RESONANT CURVED CANTILEVER WINGS FOR MICROROBOTIC FLIGHT

RESONANT CURVED PIEZOELECTRIC CANTILEVER FLUID DIODE WINGS FOR MASS-PRODUCIBLE FLYING MICROROBOTS

By MATTHEW DANIEL MINNICK, B.Eng., P.Eng.

A Thesis Submitted to the School of Graduate Studies in Partial Fulfilment of the Requirements for the Degree Doctor of Philosophy

McMaster University DOCTOR OF PHILOSOPHY (2013) Hamilton, Ontario
(Engineering Physics)

TITLE: RESONANT CURVED PIEZOELECTRIC CANTILEVER FLUID DIODE WINGS FOR MASS-PRODUCIBLE FLYING MICROROBOTS

AUTHOR: Matthew Daniel Minnick, B.Eng. (McMaster University), P.Eng. (Ontario)
m.d.minnick@gmail.com, minnick@mcmaster.ca

SUPERVISOR: Professor R.N. Kleiman

Number of pages: xxxi, 234, 187

Abstract:

This work explores a new method of force generation for flying robots on the sub-cm wingspan scale: resonant curved piezoelectric cantilevers created using completely parallel MEMS fabrication. It theorizes that because a resonating curved beam has a different drag coefficient on the upstroke than the downstroke, it should act as a fluid diode: a partial one-way gate for fluids, and thereby generate an asymmetric force over a symmetric one-degree-of-freedom flapping cycle. It develops a simplified model for the large-amplitude resonant mode of thin circular arcs by analytically extending the resonant mode shape of straight cantilevers, shows that this shape is a better fit to experimental data than previous models, and shows that it accurately predicts the resonant frequency. It uses this resonant mode to compute the force on flapping curved arcs under a wide range of amplitudes, Reynolds numbers, and arc angles using computational fluid dynamics (CFD) simulations, and extends the concept of a drag coefficient from steady-flow fluid mechanics to steady-state oscillatory fluid mechanics both for net force generation and power dissipation. It develops a framework to analyze the CFD results in the broader context of a complete robot, and uses this framework to determine priorities for material selection, robot size, and flapping shape, depending on desired robot application. It tests these theoretical predictions by creating prototype 7.6 mm wings out of 7.5 μm thick x-cut quartz and SU-8, after developing and implementing a method to smoothly thin x-cut quartz leaving the surface free of dielectric-compromising pits using reactive ion etching (RIE). Finally, it constructs a test chamber to measure the force, amplitude, and electrical parameters of the flapping wings under a variety of air pressures and demonstrates that the results are consistent with the theoretical predictions, indicating that this approach can in fact lead to successful flying microrobots.

Acknowledgements:

This project wouldn't have been possible without the support of so many wonderful people. Thank you to my supervisor Dr. Rafael Kleiman for being brave enough to let me pursue this project and enthusiastic enough to keep funding it far past the time required for it. Thank you to my PhD Committee members Dr. John Luxat and Dr. Nikola Nicolici for their sage advice and enthusiasm during committee meetings, a well as Dr. Luxat's donation of time with his simulation computer cluster and high speed camera. Thank you to my summer students Mike Burnett, Virgil McLaren, and Paul Ciolek for dedicating their time on computers and in labs carrying out work to make this project a reality. Thank you to the CEDT technical staff for their significant time teaching me how to carry out the fabrication of the physical devices, specifically Doris Stevanovic and Zhilin Peng for teaching me how to use the equipment in the clean room, and Dr. Brad Robinson and Dr. Shahram Tavakoli for teaching me how to use the MBE lab equipment, including the STS 320 RIE. Thank you to the BIMR technical staff: Dr. Anton Dabkowski for taking an interest in the project and helping me think through several technical aspects, including his help with the wafer grinding initiative; Frank Gibbs for the use of his high-precision digital scale; and especially Jim Garrett for many hours talking through technical and not-so-technical aspects of the project on top of fabricating several mechanical components of the torsion bar and shadow masks. Thank you to the CCEM technical staff: Andy Duft for his training with SEM use & sample prep lab, and trusting me enough to loan me a spare SF_6 tank when mine ran out, and Chris Butcher for his training with wafer grinding and the optical microscope. Thank you to Dr. Andy Knights for his time discussing silicon and III-V semiconductor theory. Thank you fellow MEMS grad student Abhi Rampal for being willing to discuss simulations and many hours of help with fabrication and lab operation. Thanks to fellow MEMS grad student Chris Pope for teaching me how to create PCBs for testing and help with LabVIEW programs for lock-in amplifier frequency sweeps. Thanks to fellow grad student Josh Rideout for his partnership in developing and implementing the III-V cantilever wing. Thanks to fellow grad student Gabriel A. Devenyi for much time

helping with the SEM and logistics of authoring a paper. And thank you to my dad Chris Minnick, mom & step-dad Carole & Peter O'Brien, sister Melissa, and brothers Luke, Elliott, and Aidan for their unconditional love and support.

Table of Contents:

List of Figures:

List of Tables:

List of Abbreviations and Symbols

Glossary of Abbreviations, Acronyms, and Terms

ANSYS: Suite of computer simulation programs developed by ANSYS, Inc.

BvD: Butterworth-van Dyke; an electrical model for piezoelectric resonators which views them as an RLC branch in parallel with a capacitance

Cantilever: Beam which is longer than it is wide or thick, and is rigidly supported on one end of its length

CFD: Computational Fluid Dynamics (in general)

CFX (ANSYS CFX): A commercial CFD program; part of ANSYS

COMSOL: COMSOL Multiphysics: commercial software physics simulation environment

DARPA: Defense Advanced Research Projects Agency, major funding agency in the United States with an annual budget for R&D in excess of 3 billion USD

EDS: Energy-dispersive X-ray spectroscopy

FEM: Finite element method; popular technique for numerically solving differential equations using computers

Fluid Diode: Shape or structure which allows fluid to flow by (or through) it more easily in one direction than the other

MATLAB: A commercial computer algebra system

MAV: Micro Aerial Vehicle

MEMS: MicroElectro Mechanical Systems

NAV: Nano Aerial Vehicle - DARPA defines this as a flying vehicle with wingspan smaller than 15 cm and mass lower than 20 g [1]

PCB: Printed circuit board

QSU8_17a: The name of this work's most extensively tested pair of multimorph wings made of an x-cut quartz and SU-8 stack

Quartz: Single crystal SiO_2 (by this definition, "fused quartz" is not quartz at all), in its alpha phase (which is the stable phase at STP)

RS: This work's assumed fundamental resonant mode shape for a curved cantilever presented in Section 2.2.6; RS stands for "Resonant Shape"

SEM: Scanning electron microscope

SHARCNET: The Shared Hierarchical Academic Research Computing Network (SHARCNET) is a consortium of colleges, universities and research institutes operating a network of high-performance computer clusters across south western, central and northern Ontario; www.sharcnet.ca

SU-8: UV-cured epoxy developed by MicroChem. SU-8 comes in a variety of forms; specifically, this work used SU-8 2007

List of Symbols: Variable Dictionary

Alphabetical Listing (with Greek letters after the letters that type them in Symbol font):

A = wing area = LB

A_4 = total area of all 4 wings = $4LB$

A_{r0} = Wing relative bending amplitude (relative to wing length so that tip amplitude = LA_{r0}; for details, see the RS bending mode section)

$A_{r0DCPerV}$ = Transduction of voltage signals to wing amplitudes, without resonance effects. This is the wing's low-frequency ("DC" compared to the resonant frequency) amplitude per applied volt

B = wing width (AKA chord length). $B = L/4$ is fixed to simplify the design. This ratio of chord-to-wing length is found in most insects.

c = wing curvature = 1/radius of curvature

c_0 = built-in wing curvature from strain mismatch in fabrication

C_{DC} = wing curvature amplitude

d_{piezo} = piezoelectric coupling coefficient [m/V]

$dist_{\max}$ = maximum travel range of robot between recharges.

E_{Avail} = Total energy capacity of battery (or supercapacitor)

$\varepsilon_{free0,i}$ = latent strain in each layer of the wings after processing (but not bending to their natural bending angle)

ε_r = relative electrical permittivity

F_N = net force output per wing; average force it produces over many flapping cycles.

f = frequency

f_{res} = resonant frequency

f_{resCor} = correction factor to retrieve resonant frequency of a curved cantilever using the flat cantilever formula

Φ_{DC} = DC bending angle amplitude

ϕ_0 = built-in wing bending angle (steady-state angle with no external force or voltage on the wing, due to built-in strain)

h = total wing thickness

η_{Solar} = Solar cell efficiency

$\eta_{stack} \equiv \dfrac{\rho_m^{2.5}}{\gamma_{c0}^2 E_Y^{1.5}}$: stack-based parameter to minimize in order to allow maximum Re

$\eta_{shape} \equiv \dfrac{\phi_0^2}{A_{r0}^3}$: shape-based parameter to minimize in order to allow maximum Re

I_{Solar} = ambient solar power density

L = wing length

m_{batt} = power supply mass

m_{core} = mass of the core of the robot, including the control circuitry but not the power supply or wings

m_{wings} = mass of the wings

μ_m = linear mass density = $hB^*\rho_m$

ν_{air} = kinematic viscosity of air

P_A = average power consumed by flapping a wing

P_{A2} = average power consumed by flapping 2 wings

P_{A4} = average power consumed by flapping 4 wings

$P_{Avail} = P_{Avail,Solar} + P_{Avail,Batt}$ = Total power available

$P_{Avail,Batt}$ = Power available from payload battery (or supercapacitor)

$P_{Avail,Solar} = \sigma_{Solar} A_4 = \sigma_{Solar} L^2$ = Power available from solar cells

P_{Elec} = extra electrical power losses

P_{excess} = excess power = total power available minus total power required

$P_{required}$ = total required power for flight including electrical losses

Q = quality factor

R_c = contact resistance (i.e., the electrical resistance of the electrical contacts)

ρ = wing radius of curvature

ρ_{air} = mass density of air

ρ_e = resistivity

ρ_{m} = mass density of the stack (kg/m^3); weighted average over all layers

$\sigma_{0,\mathrm{i}}$ = layer stresses during processing for layer *i*

$\sigma_{\mathrm{t,i}}$, $\sigma_{\mathrm{c,i}}$ = maximum layer tensile and compressive stresses at static bending for a layer *i*

$\sigma_{\mathrm{tres,i}}$, $\sigma_{\mathrm{cres,i}}$ = maximum layer tensile and compressive stresses at resonant bending for a layer *i*

T_{air} = ambient temperature

t_{charge} = Recharge time

t_{flight} = Flight time; Maximum time length desired for a continuous flight

t_i = thickness of wing layer #*i*

V = actuation voltage

v_{fwd} = maximum forward velocity of flying robot

VST = volume scan time; time it takes a swarm of robots made from a 4" wafer to travel 500 m, then scan a volume of 500 m^3 with a scan whose range is $1\mathrm{m} \times \sqrt{P_{Excess} / \mathrm{W}}$.

$\xi_{Shape} \equiv \frac{A_{r0}}{\phi_0}$: shape-based parameter to minimize at fixed Re for max solar power and min fluidic power required

$\xi_{Stack} \equiv \frac{\gamma_{c0} E_Y^{0.5}}{\rho_m^{0.5}}$: stack-based parameter to minimize at fixed Re for max solar power and min fluidic power required

Declaration of Academic Achievement:

I am the only graduate student who's worked on this project, and am responsible for all of its progress from launch to its current inception. I am grateful to my supervisor Dr. R.N. Kleiman for being willing to fund this research, and for the time he put into the many meetings we've had over the years and advice he's given about how to proceed towards making these wings for micro flying robots. Dr. Kleiman also funded three summer students for this project which I directed: Mike Burnett in 2007, and Virgil McLaren and Paul Ciolik in 2010.

Chapter 1 Introduction

Section 1.1 Microrobotic Flight

From the self-aware flying robot swarms of Prey by Michael Crichton to the *in vitro* nanoprobes of Star Trek, micro and nanoscale robotics is a staple future technology of science fiction. Despite what these stories may suggest, disaster is not necessarily inevitable, and flying robots the size of fruit flies have numerous helpful potential applications in reconnaissance, search and rescue, chemical monitoring, and 3D visualization.

The United States Defense Advanced Research Projects Agency (DARPA) program for nano air vehicles (NAVs) aims to develop "an extremely small, ultra lightweight air vehicle system", where "extremely small" means less than 15 cm wingspan and "ultra lightweight" means less than 20 grams of mass [1][1]. To date, very few technologies exist in this range and only on the very edge: DARPA's Hummingbird NAV, shown in Figure 1-1, has a 16 cm wingspan and 19 g mass. Still, robotic flight on this scale is sufficiently impressive that Time Magazine named DARPA's Hummingbird in its 2011 list of The Best 50 Inventions, one of only two DARPA-funded projects to make the list that year [3]. The Hummingbird is slightly more massive than the 16 g Black Hornet helicopter-type unmanned aerial vehicle (UAV) operated by the British Army, shown in Figure 1-2. Compared to the 0.3 mg mass-scale of a fruit fly ([4], [5]), there is plenty of room left to innovate.

[1] 15 cm wingspan may seem large for a "nano" aerial vehicle, but this distinguishes it from DARPA’s earlier “micro aerial vehicle” program, which included robots with 74 cm wingspans [2].

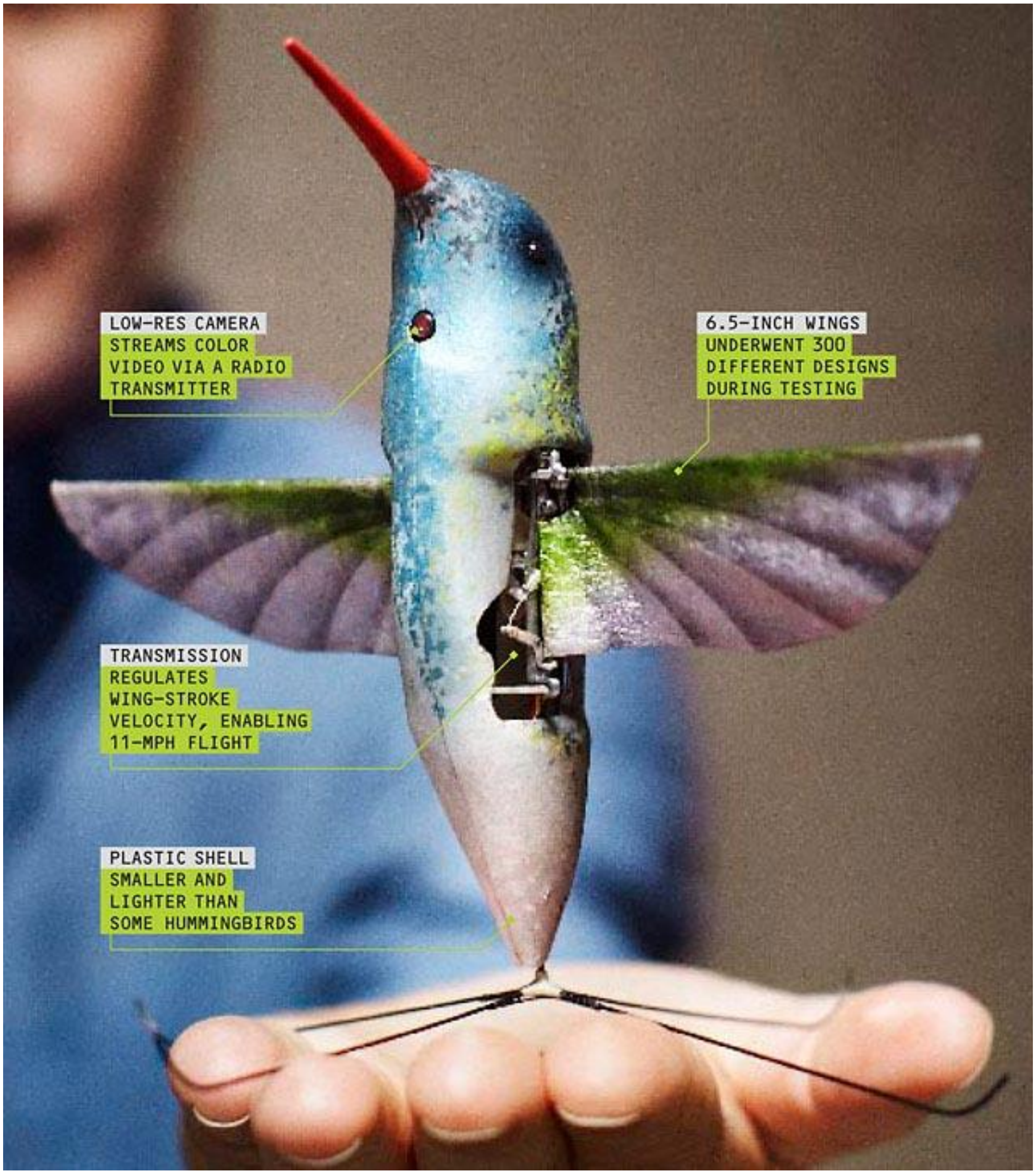

LOW-RES CAMERA STREAMS COLOR VIDEO VIA A RADIO TRANSMITTER
6.5-INCH WINGS UNDERWENT 300 DIFFERENT DESIGNS DURING TESTING
TRANSMISSION REGULATES WING-STROKE VELOCITY, ENABLING 11-MPH FLIGHT
PLASTIC SHELL SMALLER AND LIGHTER THAN SOME HUMMINGBIRDS

Figure 1-1: DARPA's hummingbird NAV [6]

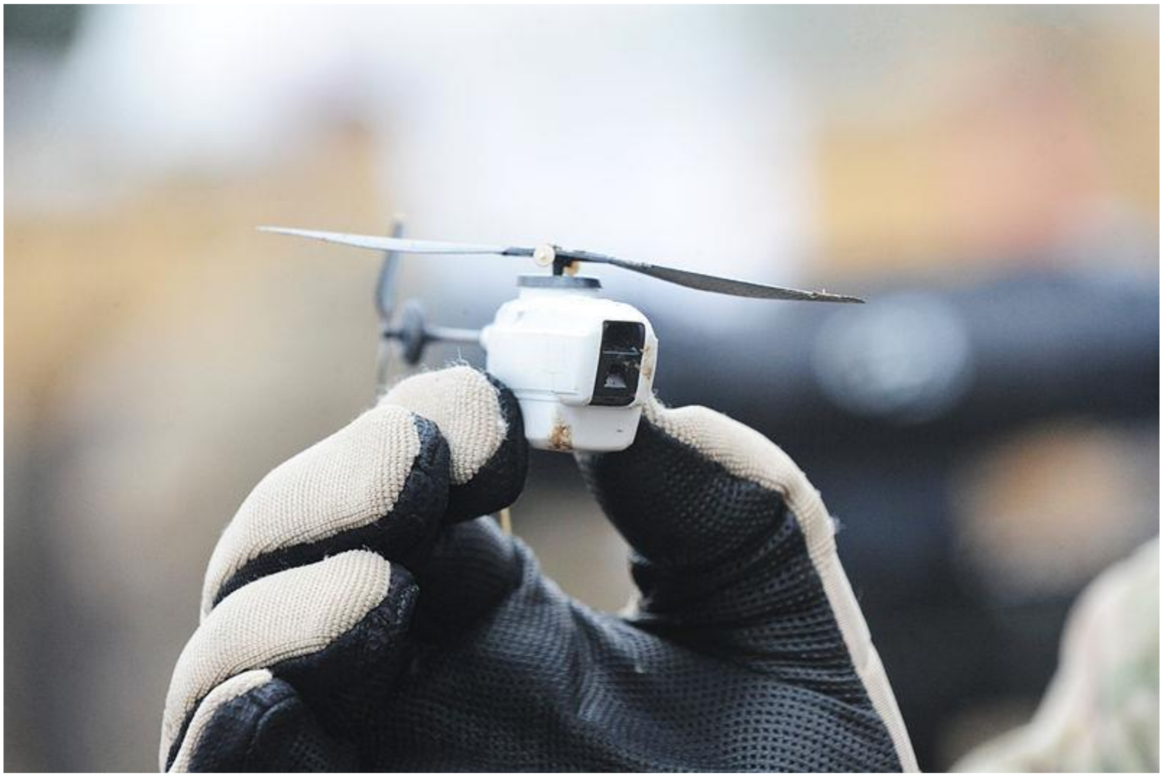


Figure 1-2: The 16 g Black Hornet UAV currently operated in Afghanistan by the British Army [7]

In fact, we may run out of names to describe small flying robots before we reach the size of a fruit fly. In 2005, DARPA's definition of an NAV was about half the size of their current Hummingbird NAV: maximum takeoff mass less than 10 g and largest dimension less than 7.5 cm [8]. Using this definition, Wood et al. extended this in [8] to define a pico air vehicle (PAV) as a flying robot with a largest dimension less than 5 cm and a takeoff mass less than 500 mg, scaling length by a factor of two-thirds and mass by a factor of one-fourth. This trend of naming is continued in

Table 1-1. To extend the table down to the 0.3 mg scale of a fruit fly requires a new SI prefix for 1000^{-9}, which for present purposes I will call ‘xeno’, after the Tsou word ‘sio’, meaning 9, and the Greek mathematician Zeno, famous for infinitely dividing distances.[2]

[2] The matching metric prefix for 1000^{+9} would then be 'xena'

Table 1-1: Classifying UAVs by metric prefix

	L	L^3	m	m/L^3
	[m]	[m^3]	[kg]	[kg/m^3]
Micro air vehicle (MAV) [8]	1.50E-01	3.38E-03		
Nano air vehicle (NAV) [8]	7.50E-02	4.22E-04	1.00E-02	2.37E+01
Pico air vehicle (PAV) [8]	5.00E-02	1.25E-04	5.00E-04	4.00E+00
Femto air vehicle (FAV)	3.33E-02	3.70E-05	1.25E-04	3.38E+00
Atto air vehicle (AAV)	2.22E-02	1.10E-05	3.13E-05	2.85E+00
Zepto air vehicle (ZAV)	1.48E-02	3.25E-06	7.81E-06	2.40E+00
Yocto air vehicle (YAV)	9.88E-03	9.63E-07	1.95E-06	2.03E+00
"Xeno" air vehicle (XAV)	6.58E-03	2.85E-07	4.88E-07	1.71E+00

Whether or not we can decide what to call them, fruit-fly sized robots are far removed from the fliers of today - but why is this? Any non-tethered flying robot intended to work in the dark would need 1) an energy storage mechanism (i.e., a super capacitor or a battery). The robot would also need 2) thrusters (e.g., wings) to turn this energy into a flight force. In order to know where it is and be able to scout its environment, the robot would also need 3) sensors (e.g., cameras, accelerometers, gyroscopes, and specific job-related sensors, such as chemical sensors). It would likely also need 4) wireless communications capability to interact with its control centre (and possibly other robots). Finally, it would need 5) computer control circuitry to manage all of this. Of these five areas, all except for the thrusters are already active fields of research for applications unrelated to micro flying robots. Therefore, research on the thrusters, the physical mechanism causing flight, would have the biggest impact toward their realization.

Many groups are currently researching flight mechanisms for extremely small microrobots. Of these, most are pursuing copying insect flapping, including the incredibly small Harvard RoboBee, shown in Figure 1-3 with its 80 mg mass and 3 cm wingspan which, as of May 2013, can now hover and land with power provided by small wires [9]. This is a huge accomplishment in the field - demonstrating hovering for a known mass determines the force output of the wings and thus their effectiveness with various power supplies attached.

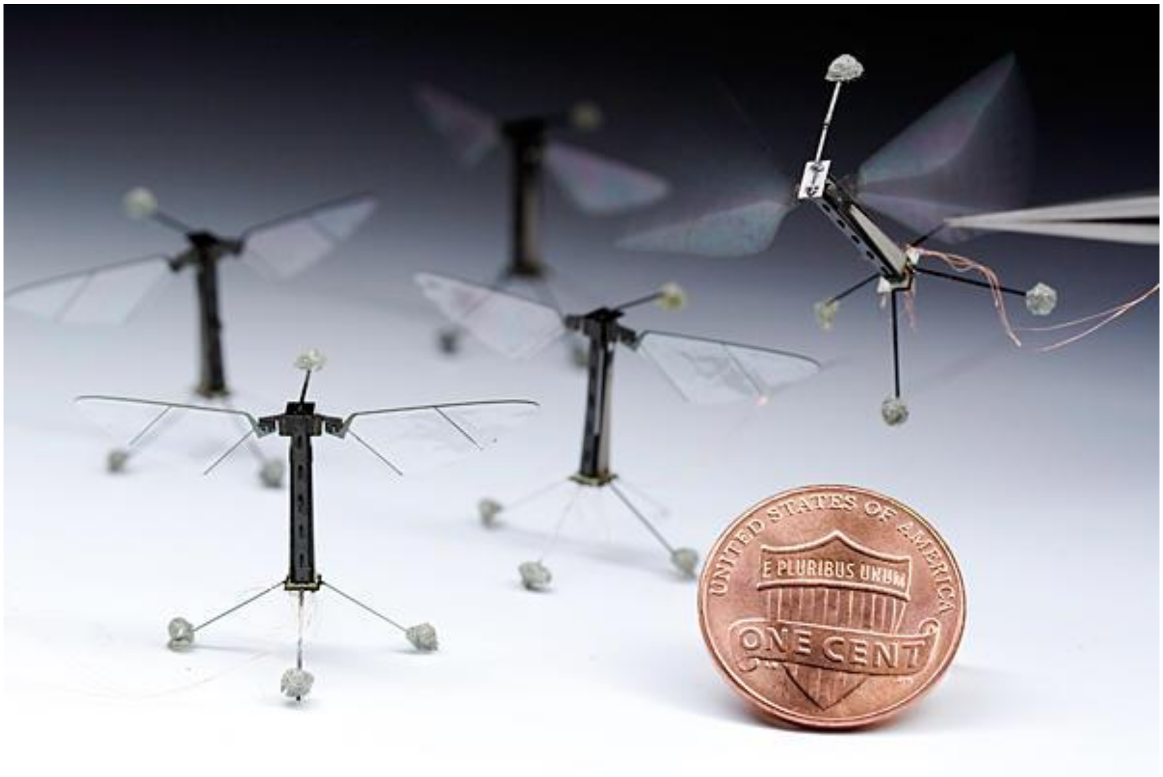


Figure 1-3: The biomimetic Harvard RoboBee has an 80 mg mass and can lift itself and land with power provided by wires [10]

As design sizes scale down further and the number of robots manufactured from a single chip increases, every serial fabrication step becomes relatively more costly making parallel fabrication increasingly important. Figure 1-4 shows the RoboBee's fuselage assembly structure which uses origami to facilitate mass-production and self-assembly, allowing parts to be manufactured in parallel processes, and individually snapped together afterwards [11]. A completely parallel MEMS fabrication approach would represent a large cost-savings for mass-produced microrobots.

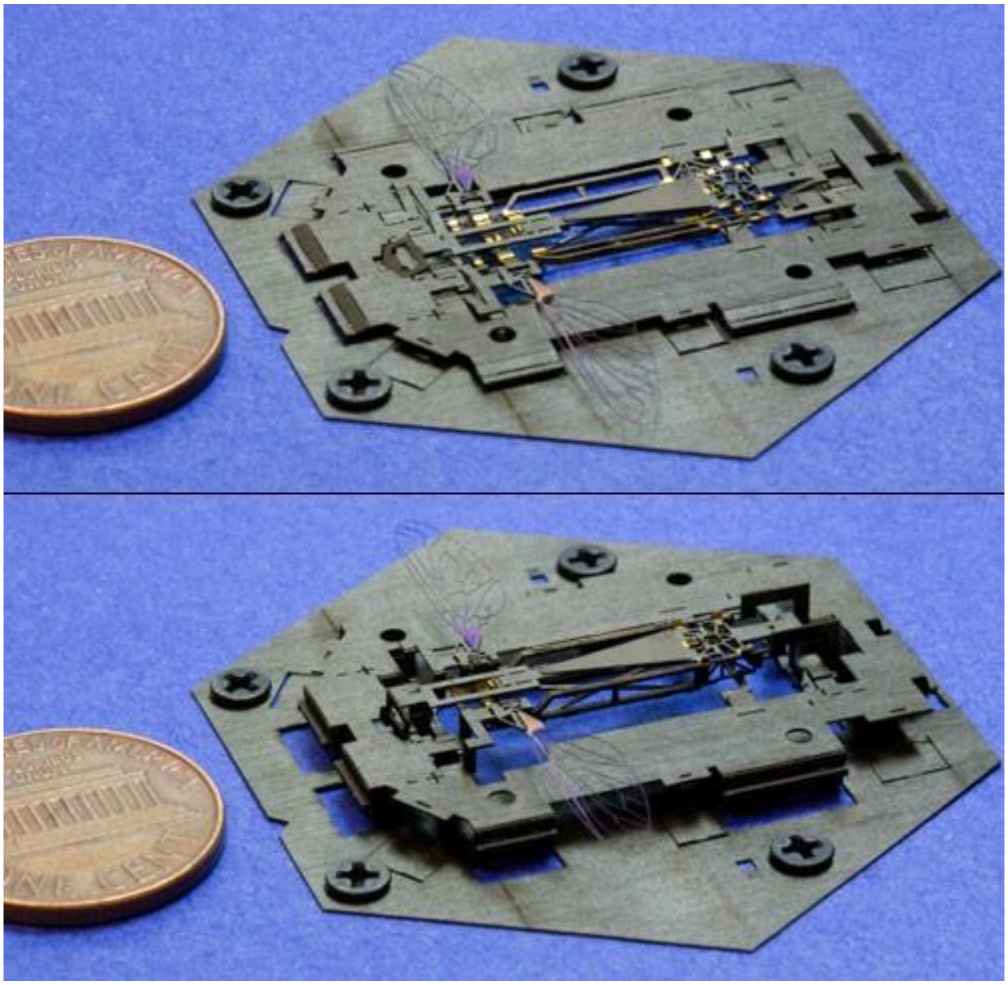


Figure 1-4: The fuselage assembly structure of the Harvard RoboBee facilitates mass-production by using origami folding techniques [11]

To date, self-powered autonomous flight on the scale of the RoboBee is unheard of, and even force measurement is very rare. In fact, it is a publish-worthy achievement to build a structure on this scale that can produce similar frequencies and/or stroke amplitudes as insects, especially using MEMS fabrication [12], [13], [14].

But what if looking at how much better insects are than our current robots is exactly what is limiting us? Sure, insects are great fliers - but is there another way to fly that, while not as optimal as nature's solution, is easier to build and nevertheless works? Indeed, the neurons of the human brain have a lot of advantages over transistors, but that certainly doesn't mean that computers based on transistor technology aren't useful.

So then, what *other* ways could we generate force besides the way an insect does it? For that matter, how *does* an insect fly?

Section 1.1.1 Flight Mechanisms

Across natural and man-made objects, there are two general approaches for flight, which I classify as "lift flight" and "thrust flight".

Lift Flight

Large scale fliers like most birds tend to use an approach similar to planes: having large wings and relying on a forward motion (provided by a thrust force) and complex wing shape to generate a lift force which opposes gravity, as shown in Figure 1-5. Since the lift force relies on forward velocity rather than forward force, lift flight fliers can also glide: allow lift to balance weight for a time with no thrust force, as shown in Figure 1-6.

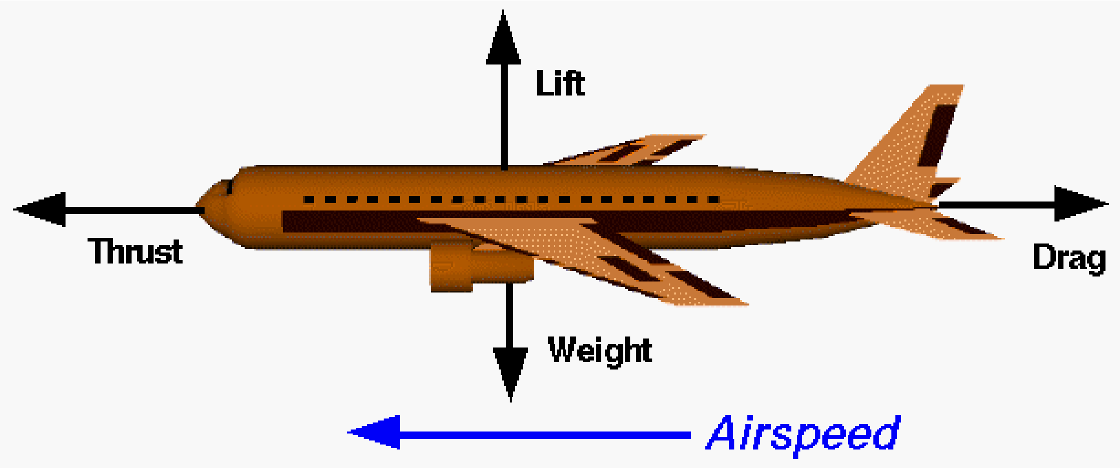


Figure 1-5: Force diagram for an airplane; a lift flight flier [15]

Figure 1-6: An eagle gliding using lift-flight [16]

Thrust Flight

In contrast, smaller scale fliers like hummingbirds and insects tend to use flapping to produce a thrust force directly opposing gravity, as shown in Figure 1-7. These fliers are characterized by their ability to hover, but must constantly keep flapping to maintain altitude (Figure 1-8). Helicopters and rockets use a similar flight approach by this definition.

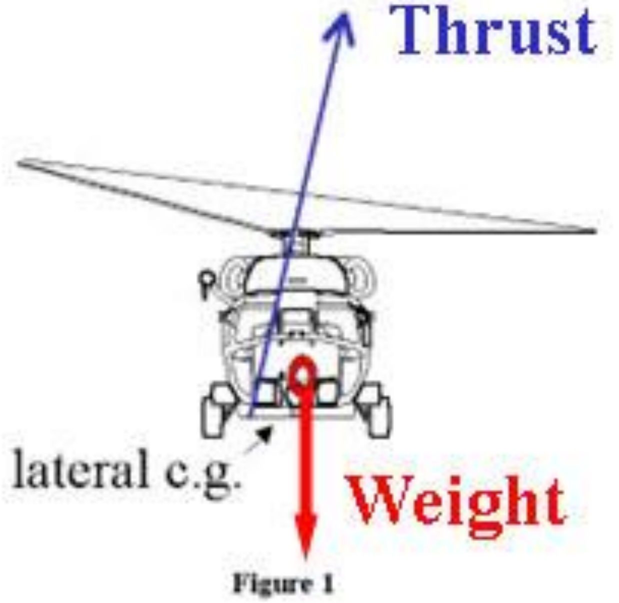


Figure 1-7: Force diagram for a helicopter; a thrust flight flier (adapted from [17]).

Figure 1-8: Hummingbird hovering; an example of thrust flight [18].

So which strategy is best for a robot on the fruit fly scale? Given that smaller species favour thrust flight, perhaps this motif is superior for smaller scales, but can we prove this? In Appendix A, I calculate that the required power for flight scales with wing length L in different ways for both types of flight. For thrust flight, required power scales with $L^{3.5}$ for large sizes, and with L^{2} for small sizes. For lift flight, required power scales with $L^{3.5}$ on the largest sizes, $L^{1.25}$ for intermediate sizes, and $L^{0.5}$ for the smallest sizes. On the smallest sizes then, miniaturizing a flyer to 1/4th its length results in 1/16th the required power for thrust flight, but only half the required power for lift flight. It is not surprising then that thrust flight dominates for nature's smallest fliers. Figure 1-9 plots the ratio of available to required power vs. wing length for the two flight mechanisms. Indeed, thrust flight dominates for sufficiently small sizes.

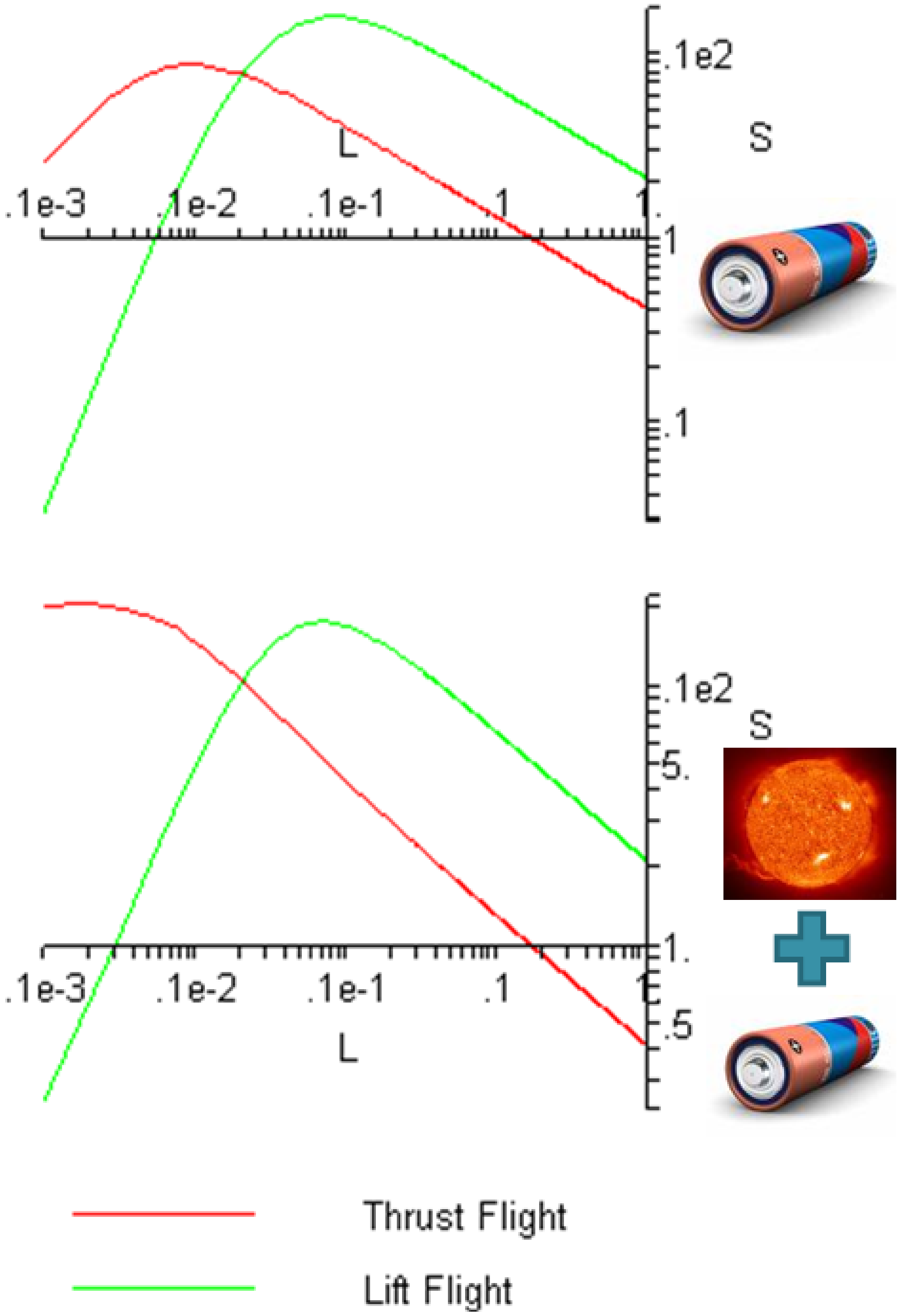


Figure 1-9: 0th-order model ratio of available to required power for thrust and lift flight for battery power alone (top) and battery power + solar power (bottom), length in meters

Section 1.1.2 Type of Thrust Mechanism

Even within thrust flight there are several different strategies:

1. Rotary motion,
2. Rocket propulsion,
3. 3 Degree of Freedom (DoF) flapping, and
4. 1 DoF flapping.

Rotary motion involves using propellers to generate a net force. This is the thrust flight strategy of helicopters, but is not very suitable for small scales because of friction and very difficult fabrication (Figure 1-10).

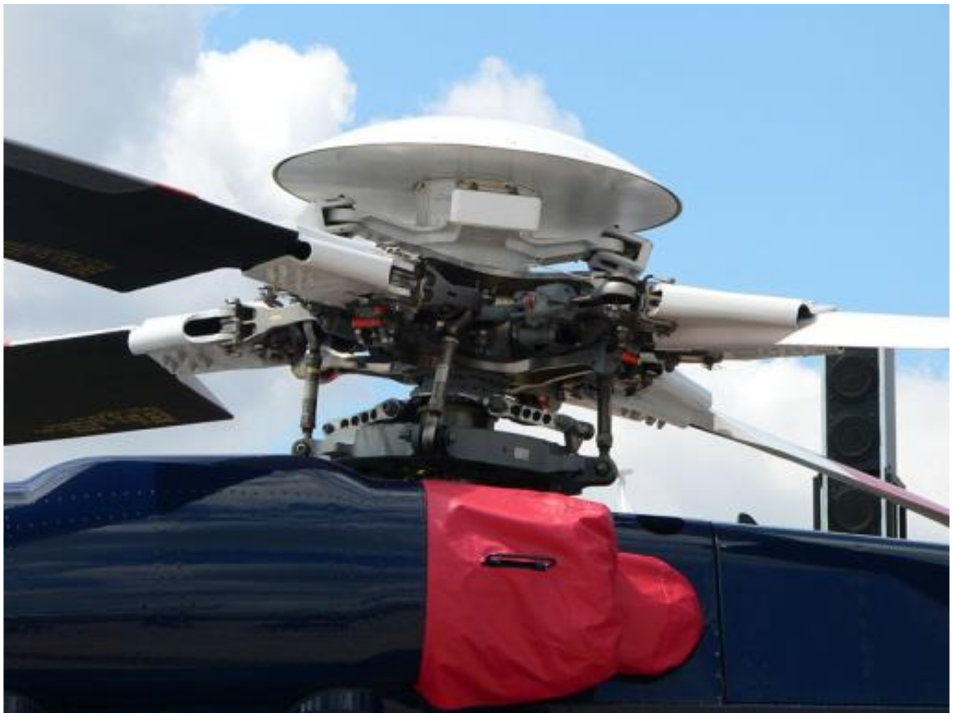


Figure 1-10: Complexities of a helicopter rotor [19]

Some work on micro rocket propulsion has been done at MIT and Berkeley [20], but this is not suitable for the kind of continuous flight a robot would require.

3 DoF flapping means moving wings in a complicated shape in three dimensions. Insects manage to do this passively ([21], [22]), but building an actuator like that on the same scale is extremely complicated and certainly beyond my abilities. As mentioned before, there are several very talented teams working on this elsewhere.

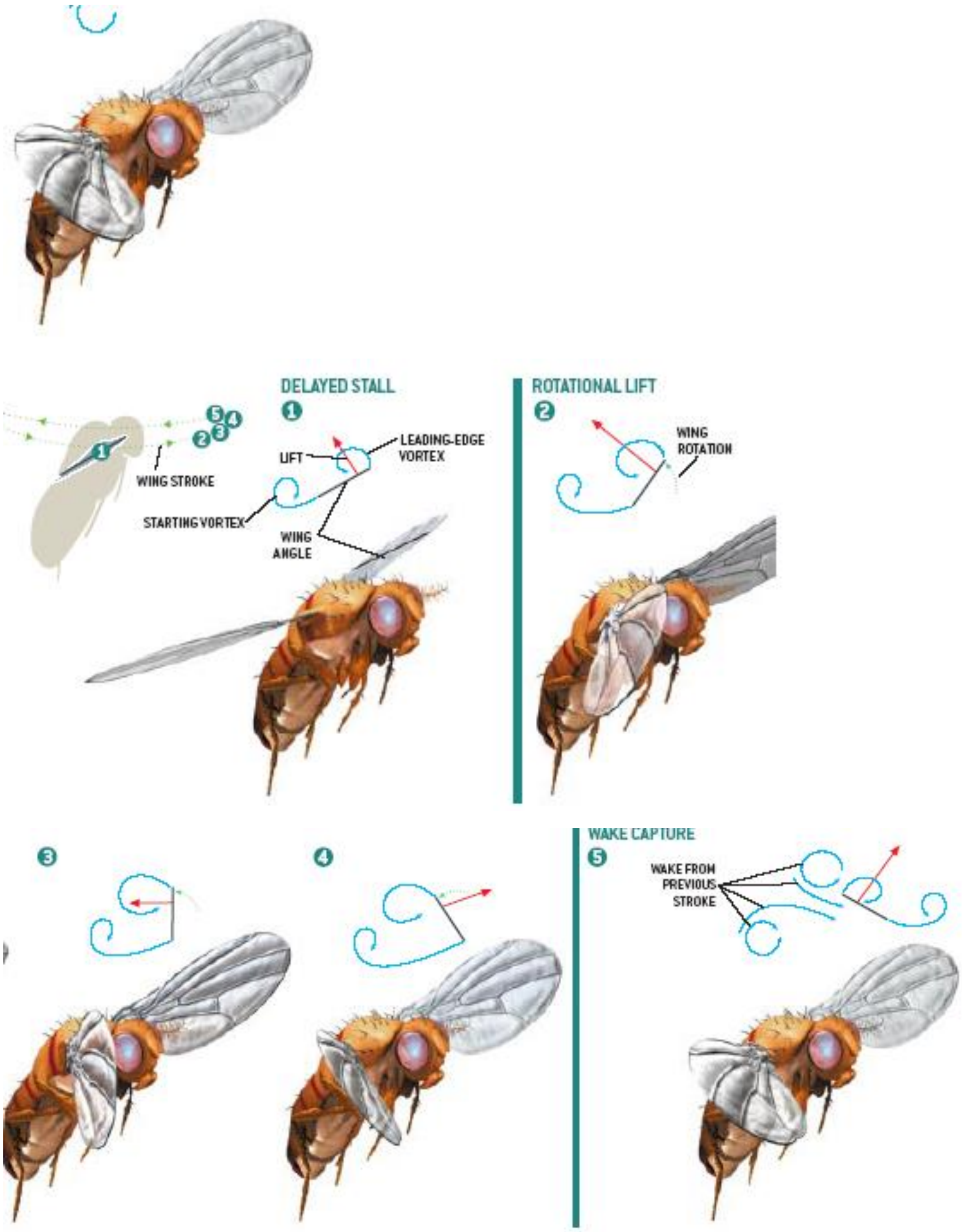


Figure 1-11: 3 DoF flapping motion of a fruit fly, from [23].

1 DoF flapping means moving the wings up and down along the same path, generating a net force due to some asymmetry in the shape of the wings themselves. For instance, consider the wings from Figure 1-12. These wings have a rigid support structure that allows the flexible flap to release from the wings on the upstroke, but not on the downstroke. Essentially, these wings are "fluid diodes" (analogous to electrical diodes): they allow fluids (i.e., air) to pass through them more easily in one direction than the other.

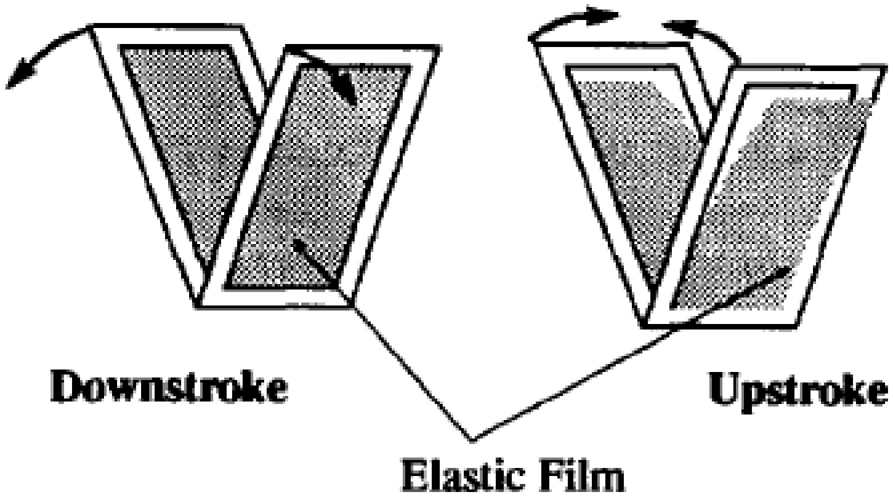


Figure 1-12: Fluid diode wing using one large flap as envisioned by Kubo et. al. in 1993 [24].

Fluid diode wings have also been proposed using a single large flap on a rigid grid, as in Figure 1-13. This approach makes it clear why the flap can only open in one direction.

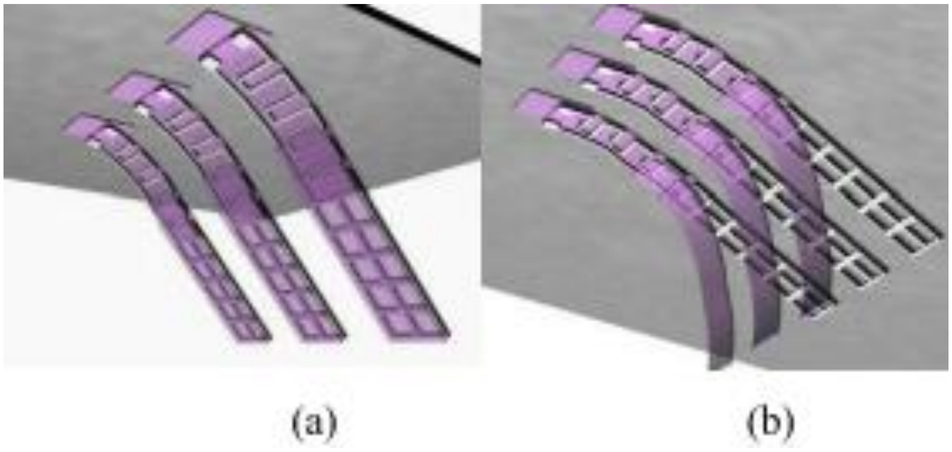


Figure 1-13: Fluid diode wings proposed by Chan et. al. in 2004 [13].

Both Kubo in 1994 and Chan in 2004 managed to flap the structures but did not demonstrate a net force from them ([24], [13]). Past studies on fluid diode wings have produced potentially functional wings and flapped them at frequencies close to the ones that insects use, but have not demonstrated any net force, let alone specifically measured it. Managing to do so would be a significant contribution to the current field of flying microrobots.

Section 1.2 Basic robot design for 1 DoF Flight

Wings using 1 DoF flapping require simple fabrication compared to those using 3 DoF flapping. However, 1 DoF flapping is not actually seen in nature, which suggests that it is inferior to 3 DoF flapping. Further, 3 DoF flapping designs entail enough control to steer with only 2 wings, but 1 DoF flapping does not have this option because each wing can only control the magnitude but not the direction of its thrust. Nevertheless, it may be possible to build robots that fly with 1 DoF flapping using at least 4 wings.

Consider Chan et al.'s robot design, shown in Figure 1-14. By reducing the force output of the front wings, the robot will tilt and part of its thrust force will resolve to accelerate the robot forward through the air.

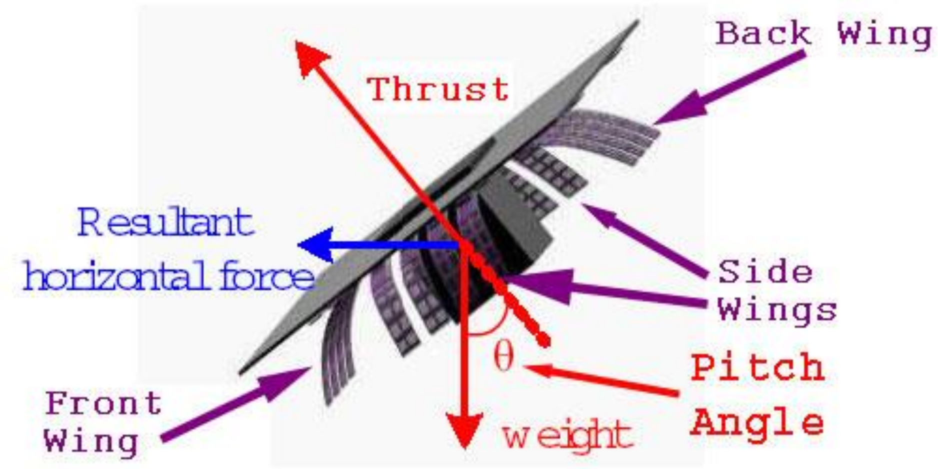


Figure 1-14: Chan et al. 2004's robot design [13], labeled and displaying force vectors.

This would also work with the wings in an H-pattern rather than an X-pattern, as shown in Figure 1-15. This robot design consists of curved piezoelectric unimorph wings, a central fuselage for control, and a hanging power supply for pendulum-action stability. The H-pattern is necessary because the piezoelectric effect in quartz couples vertical electric fields (those in the crystal x-direction) to only one of the transverse directions (the crystal y-direction).

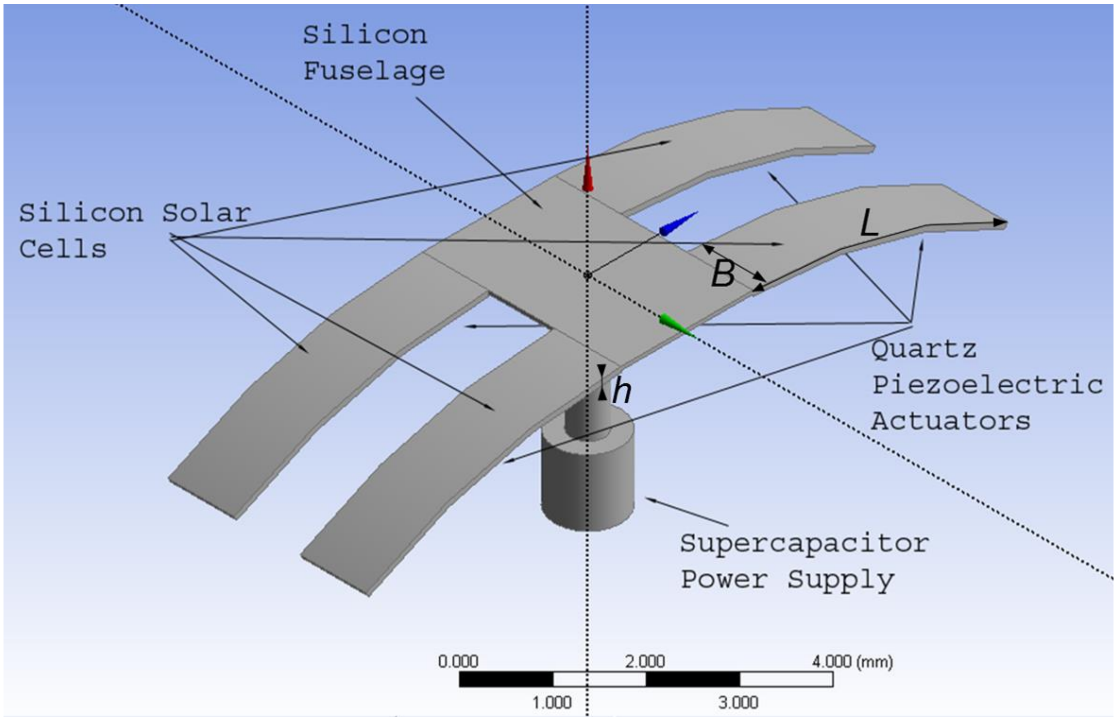


Figure 1-15: Robot Design using Curved Cantilever Wings, modelled in ANSYS

In either design, the force output to any one wing can be adjusted by changing the voltage amplitude or duty cycle of the signal applied to that wing, and by changing the relative force output of the front vs. back wings, the robot can tilt and subsequently move forward.

Section 1.3 Other Fluid Diode Wings

A flexible flap on a static grid may not be the only way to build a fluid diode wing. The concave side of a hemisphere has a significantly larger drag coefficient than its convex side Figure 1-16. This means that if a hemisphere moves up and down in a fluid, on average, it will generate a net force.

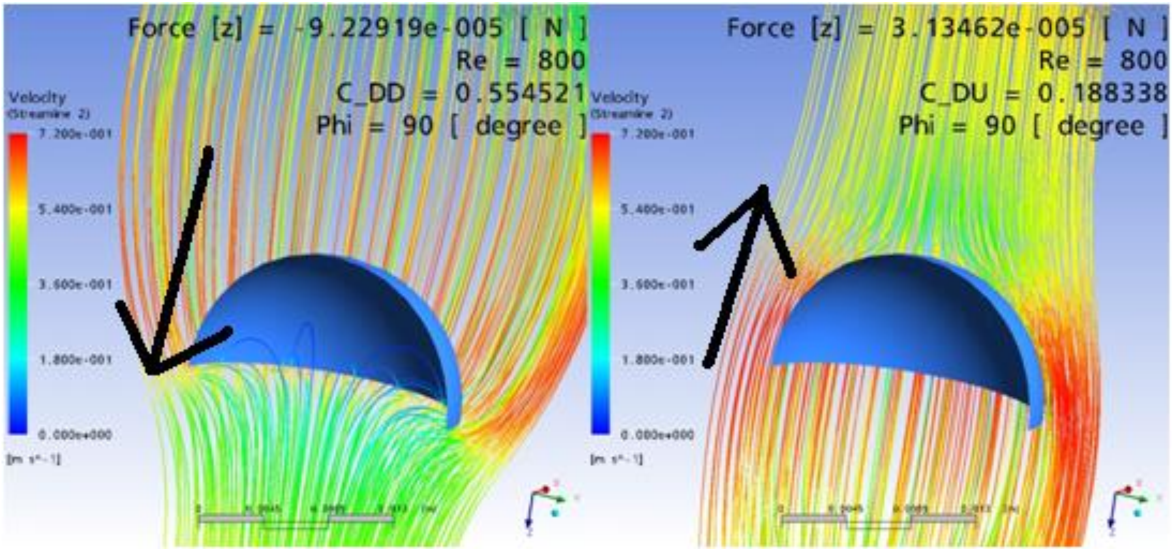


Figure 1-16: Hemisphere moving down shows a higher drag coefficient in steady state (0.55) than when moving up (0.19)

Now, consider a curved cantilever in its fundamental resonance, which causes it to open and close, as shown in Figure 1-17. Since on average the shape is a curved hemisphere section, would it too generate a net force?

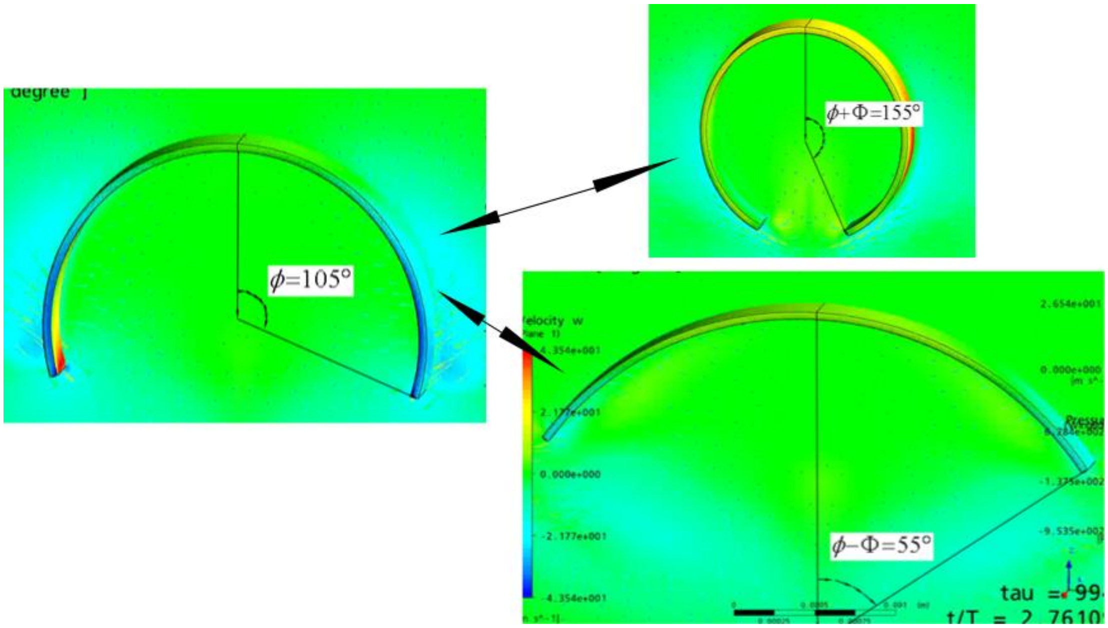


Figure 1-17: An opening and closing curved cantilever with arc-angles labeled. This is another example of a fluid diode wing.

After a series of simulations of opening and closing cantilevers the answer seems to be yes. Even though a resonating curved cantilever sweeps back and forth along the same path, its fluid diode properties generate a net force in the process.

But would the net force from a resonant curved cantilever be enough for flight? There is a lot of information needed to answer this question. For one, we need to know the large-amplitude resonant shape of curved structures. Next, we need the force generated by flapping in this manner, and the power dissipated in the process. Furthermore, we need a way to evaluate whether the force generated and power dissipated would lead to a successful robot.

To solve these problems, I fabricated thin curved cantilevers which I could resonate. To find a model for the flapping shape, I analytically extended the resonant mode of a flat cantilever to curved cantilevers, and showed that it matched the resonant shape of the fabricated structure. To determine the net force and dissipated power, I carried out tens of thousands of computational fluid dynamics (CFD) simulations and compared these to the measured net force and power amplitude of the fabricated structure. To determine whether this would lead to a successful robot, I developed an analytical framework for evaluating resonant wings in the context of a complete robot, and demonstrated that this flapping actually *can* lead to successful flying robots.

To summarize, this thesis is about resonant piezoelectric cantilever fluid-diode wings for monolithically-fabricated NAVs on the fruit fly scale, where:

1. "monolithically fabricated" means built using parallel processing techniques (as in the integrated circuit industry) rather than assembly in a serial manner,
2. "NAV on the fruit fly scale" means a flying robot with a wing span of 2 mm - 2 cm, and a mass on the order of 1 mg,
3. "resonant piezoelectric cantilever" means the wings are stacks of piezoelectric layers (and possibly passive layers) which move in response to an applied voltage,

and are operated at resonance (specifically, their fundamental transverse resonant mode), and

4. "fluid diode wing" means a wing which allows air to pass by (or through) it more easily on the upstroke than the downstroke, and so generates an asymmetric force from a symmetric flap cycle.

Section 1.4 Overview of Contribution

This thesis project has made a number of contributions towards realizing this type of NAV:

1. Determined the large-amplitude resonant shape of a curved cantilever.
2. Used CFD simulations to determine the force output of wings flapping with this resonant shape of a curved cantilever (i.e., of the shape found in step 1), and the power they dissipate in the process, over a large range of three parameters: Reynolds number, flapping amplitude, and initial curvature.
3. Created a complete big-picture framework to combine the force and power data from the CFD simulations with a piezoelectric stack design and determine useful information about the NAV each setup would lead to in order to determine both scaling rules for robot design and actual predicted robot specifications.
4. Developed a method to selectively thin x-cut quartz wafers to < 10 μm thickness using shadow masking and a standard parallel-plate (non-ICP) reactive ion etching (RIE) system, with a sufficiently smooth surface to be electrically insulating and usable as robot wing sections.
5. Fabricated curved wing structures with x-cut quartz and SU8 for testing theory predictions.
6. Tested quartz:SU8 wings and demonstrated that the curvature, resonant frequency, resonant shape, flapping amplitude, electrical characteristics, and net force they produce are consistent with the predictions from the CFD simulations and analytical framework, including the scaling across different gas pressure ranges.

7. Devised several other possible wing structures and calculated the qualities of the robots they would lead to in order to map out potential future directions for this field.

This thesis will present these contributions in this order, explaining background information and theory as-needed.

Chapter 2 Curved Cantilever Resonance

A cantilever is a beam with one dimension longer than the other two and one end of the long dimension rigidly clamped. Resonant analysis of straight cantilevers is well established in literature (e.g. [25]), but descriptions of curved cantilever resonance are surprisingly sparse. Curved cantilevers result whenever different layers have a strain mismatch after fabrication. While this is usually undesirable in MEMS, in this project it is useful: curved cantilevers act as fluid diodes and so are potentially usable as microrobot wings. To design these wings, we need information about cantilevers:

1. What natural bending angle (i.e., bending angle without any externally-applied force or voltage) will a given cantilever have?
2. What is the resonant frequency of a curved cantilever?
3. What is the resonant shape of a curved cantilever?

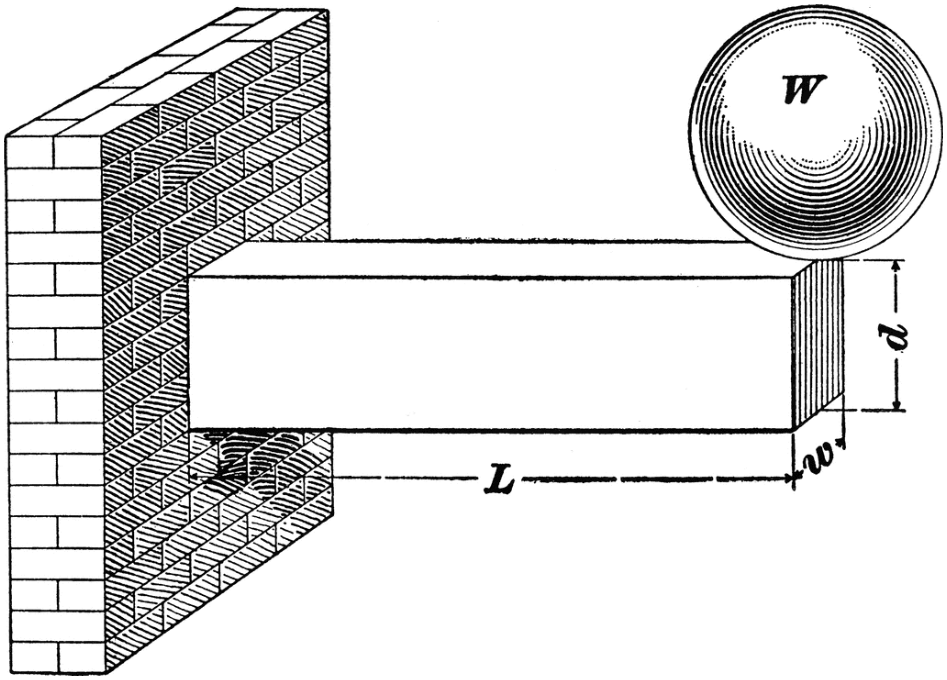


Figure 2-1: A cantilever with length *L*, width *w*, and thickness *d* supporting a weight *W* at its tip, from [26]

In this chapter, I collect and present important equations on cantilever curvature and extend the math of straight cantilevers to determine the resonant shape and frequency of curved cantilevers.

Section 2.1 Introduction: Bimetal Strip Curvature

Suppose you bond two flat thin metal strips with different thermal expansion coefficients and then change the temperature. One layer wants to expand more than the other does, and the result is a curved structure, as shown in Figure 2-2.

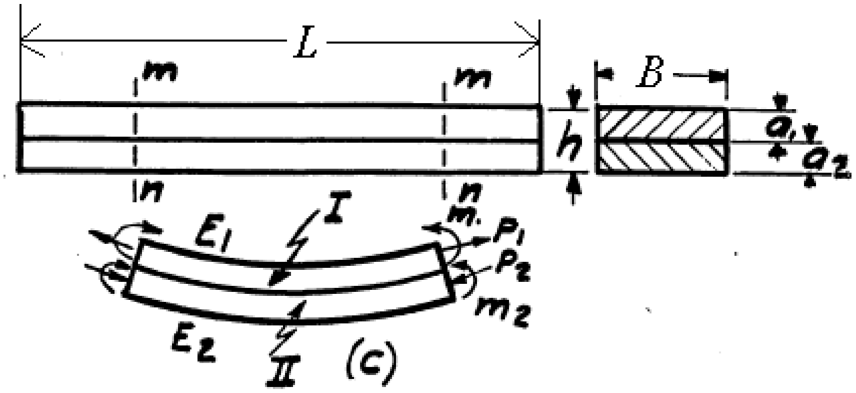


Figure 2-2: Variable numbering for layers in a bimorph; taken from [27] and modified to show L & B.

In [27], "Analysis of Bi-metal Thermostats", 1925, Timoshenko shows that the curvature ($c = 1/\rho$, where ρ is the radius of curvature) is related to the longitudinal thermal strains in its top (#1) and bottom (#2) layers via:

$$\frac{1}{\rho}=\frac{6\left(\varepsilon_{1\_2}-\varepsilon_{1\_1}\right)\left(1+m\right)^2}{h\left[\left(m^2+\frac{1}{nm}\right)\left(1+nm\right)+3\left(1+m\right)^2\right]}$$

where $m\equiv\frac{a_1}{a_2}$ is the ratio of thicknesses and $n\equiv\frac{E_1}{E_2}$ is the ratio of Young's moduli, and the strains are ε_{1_2} & ε_{1_1} (the first subscript on the strains refers to the direction, in this case, 1 = x, while the second refers to the layer: 1 = top, 2 = bottom).

Strain is the change in length per unit length: $\varepsilon = \frac{L - L_0}{L_0}$. Where the strains are thermal, as in Timoshenko's analysis, $\varepsilon = \alpha \Delta T$ (where α is the linear thermal expansion coefficient and ΔT the change in temperature), and so:

$$\frac{1}{\rho} = \frac{6(\alpha_2 - \alpha_1)(t - t_0)(1+m)^2}{h\left[\left(m^2 + \frac{1}{nm}\right)(1+nm) + 3(1+m)^2\right]}$$

The piezoelectric effect is another source of longitudinal strains. For the popular piezoelectric material lead zirconium titanate (PZT) (or any material with a nonzero d_{31} coupling coefficient), a field E_3 in the z-direction causes a proportional strain $\varepsilon_1 = d_{31}E_3$ in the x-direction. By replacing thermal strains with piezoelectric ones, Timoshenko's formula can just as easily calculate the curvature of a piezoelectric bimorph:

$$\frac{1}{\rho} = \frac{6\left(d_{31\_2}E_{3\_2} - d_{31\_1}E_{3\_1}\right)(1+m)^2}{h\left[\left(m^2 + \frac{1}{nm}\right)(1+nm) + 3(1+m)^2\right]}$$

Substituting applied voltages in place of electric fields, this becomes:

$$\frac{1}{\rho} = \frac{6\left(d_{31\_2}\frac{V_2}{a_2} - d_{31\_1}\frac{V_1}{a_1}\right)(1+m)^2}{h\left[\left(m^2 + \frac{1}{nm}\right)(1+nm) + 3(1+m)^2\right]}$$

If the bimorph is symmetric and poled so that $d_{31\_1} = d_{31\_2}$ then you should apply oppositely directed fields for maximum bending angle. This is an advantage over thermal bimorph designs: with the piezoelectric effect, it is possible to have one layer compress while the other extends to enhance the resultant curvature.

Because there are two active layers, the structure described above is commonly called a "bimorph". For the piezoelectric cantilever, it also is possible to have only one active layer, the other being nonpiezoelectric. In that case, the stack is called a "unimorph", and the curvature is given by the same formula but with one of the voltages equal to 0:

$$\frac{1}{\rho}=\frac{6\left(d_{31\_2}\frac{1}{a_2}\right)(1+m)^2}{h\left[\left(m^2+\frac{1}{nm}\right)(1+nm)+3(1+m)^2\right]}V_2$$

In any case, the curvature is proportional to $d_{31}E/h$, and the proportionality depends on the two layers' thickness ratio *m* and stiffness ratio *n*. This proportionality,

$\dfrac{(1+m)^2}{\left(m^2+\frac{1}{nm}\right)(1+nm)+3(1+m)^2}$, is plotted in Figure 2-3.

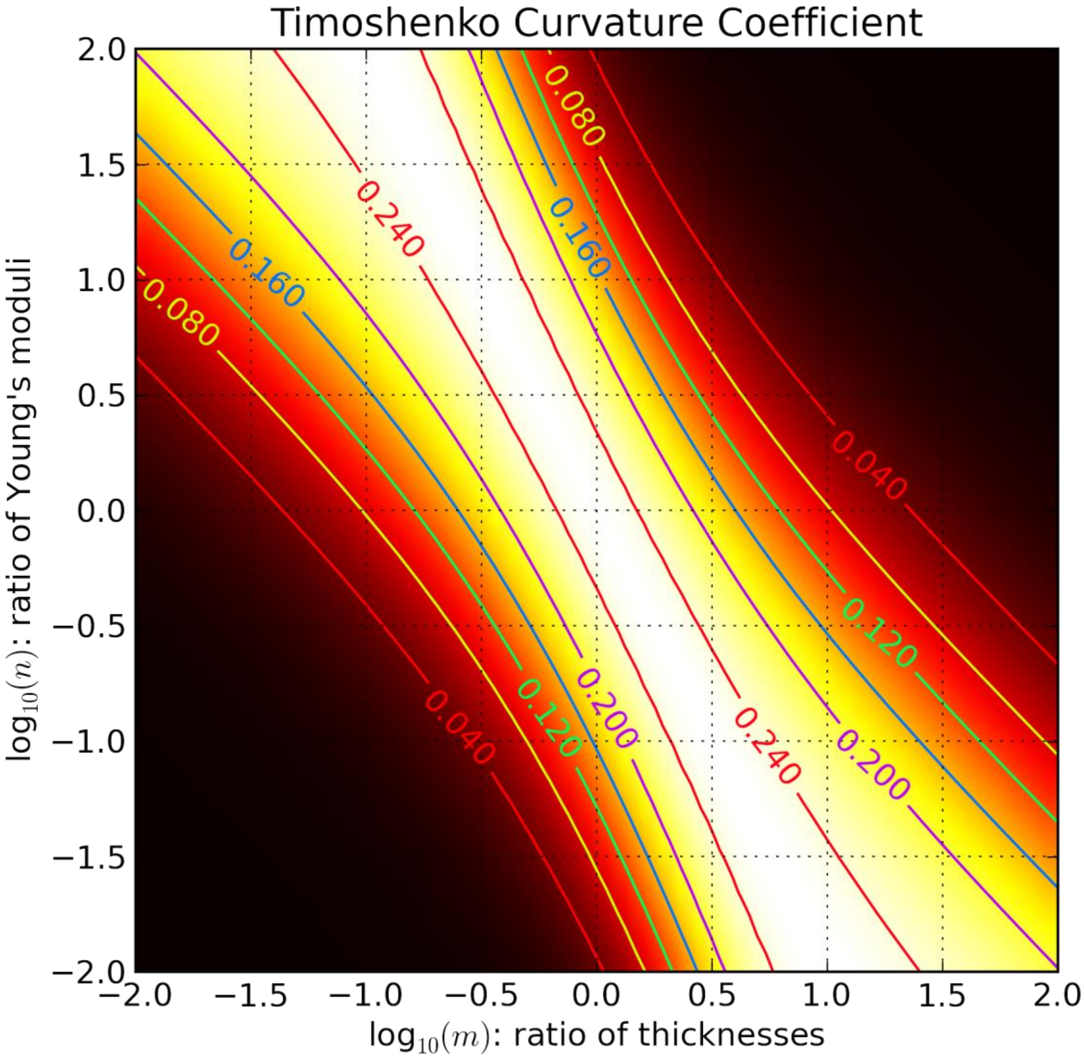


Figure 2-3: Timoshenko Curvature Coefficient

The maximum occurs around the line $\log n = -2\log m$ → $m = \frac{1}{n^2}$. This formula is how to maximize curvature for a fixed pair of materials, electric field, and thickness. E.g., if the top layer is made from a material that is twice stiff as the bottom layer (*n* = 2), then it should be only one quarter the thickness ($m = \frac{1}{n^2} = \frac{1}{4}$) in order to maximize curvature for a given strain.

Unfortunately, curvature is not the only thing that changes when changing materials and their relative thicknesses. For instance, material choice has a big effect on the mass of the wing, which affects robot weight, energy stored per displacement, and a number of other important robot parameters in subtle ways. This illustrates the need to consider the wings in the context of a complete robot when making wing design choices. This formula is a good start, but the framework in Chapter 4 does a much better job of material and thickness selection when considering the big picture of the robot as a whole by considering the effects of more than two layers.

Section 2.2 Theory of Cantilever Statics

The Timoshenko formula works for a two-layer cantilever, but what about a cantilever with three or more layers? Further, how does the cantilever's curvature translate into the angle the cantilever bends through, and the displacement of any point on the beam? What stresses and strains manifest in various beam layers? This section presents the answers to these more detailed cantilever statics questions.

Section 2.2.1 Curvature (and Bending Angle)

Curvature c is the inverse of the radius of curvature ρ: $c = \frac{1}{\rho}$. By definition, *positive* radii of curvature cause wings to bend *up*.

Curvature is related to wing length L and bending angle ϕ via the arclength formula:

$$L = \phi\rho \leftrightarrow c = \frac{1}{\rho} = \frac{\phi}{L}$$

In reality, radius of curvature (and hence curvature) varies through the thickness of the material, but for typical dimension ratios for microrobotic wings, this variance is small, and can be neglected in certain calculations. For example, an L = 8 mm long wing with a thickness of h = 20 μm and a bending angle of $\phi = \pi/2$ will have a radius of curvature of

$\rho_0 = L/\phi = 5.093$ mm at the centre, while the wing bottom has $\rho_{\text{bottom}} = 5.103$ mm, and the wing top has $\rho_{\text{top}} = 5.083$ mm.

Curvature of a beam depends on the layer materials and thicknesses, and the strains in the layers. I distinguish between the static, DC "amplitude", and resonant amplitude curvature.

1. c_0, Static curvature (AKA precurvature): the curvature present with no voltage signal or external stress.
2. C_{DC}, DC "amplitude" curvature: the extra curvature beyond the static curvature produced by a DC or low frequency voltage signal of a given magnitude. $C_{DC} = c_{DC} - c_0$.
3. C or C_{res}: Resonant amplitude curvature: the amplitude of the curvature produced by a voltage signal at the wing's resonant frequency. $C_{res} = QC_{DC}$, where Q is the quality factor.

Often we will use bending angle rather than curvature, but the same nomenclature applies:

1. $\phi_0 = c_0 L$: static bending angle
2. $\Phi_{DC} = C_{DC} L$: DC bending angle "amplitude"
3. $\Phi = \Phi_{res} = Q\Phi_{DC}$: resonant bending angle amplitude

Section 2.2.2 Multimorph Curvature

In reality, piezoelectric stack actuators are rarely truly unimorphs or bimorphs, because the electrical contacts have nonzero thickness and so affect the mechanical properties of the stack. A piezoelectric stack with an arbitrary number of active and passive layers is in general known as a "multimorph".

Multimorph curvature was explained by DeVoe et al. in [28], and is summarized below.

Curvature for a multimorph is:

$$c \equiv \frac{1}{\rho} = \frac{\mathbf{DA}^{-1}\mathbf{C}}{1 - \frac{1}{2}\mathbf{DA}^{-1}\mathbf{B}}$$

where $\mathbf{D} = \frac{1}{\sum_{i=1}^{m} E_i I_i}\begin{bmatrix} \frac{t_1}{2} & t_1 + \frac{t_2}{2} & \cdots & \sum_{j=1}^{m-1} t_j + \frac{t_m}{2} \end{bmatrix}$

$$\mathbf{A} = \begin{bmatrix} \frac{1}{A_1E_1} & \frac{-1}{A_2E_2} & 0 & \cdots \\ 0 & \frac{1}{A_2E_2} & -\frac{1}{A_3E_3} & \ddots \\ 0 & 0 & \ddots & \\ \cdots & 0 & \frac{1}{A_{m-1}E_{m-1}} & \frac{-1}{A_mE_m} \\ 1 & 1 & 1 & 1 \end{bmatrix}, \mathbf{B} = \begin{bmatrix} t_1 + t_2 \\ t_2 + t_3 \\ \vdots \\ t_{m-1} + t_m \\ 0 \end{bmatrix}, \ \mathbf{C} = \begin{bmatrix} \varepsilon_{free\_2} - \varepsilon_{free\_1} \\ \varepsilon_{free\_3} - \varepsilon_{free\_2} \\ \vdots \\ \varepsilon_{free\_m} - \varepsilon_{free\_m-1} \\ 0 \end{bmatrix},$$

Layers of the stack are numbered from bottom to top, positive *c* indicates an upward curvature, A_i is the cross sectional area of the *i*th layer ($A_i = Bt_i$), t_i is the thickness of the *i*th layer, E_i is the Young's modulus of the *i*th layer, $\varepsilon_{induced\_i}$ is the total induced strain (piezoelectric plus thermal) in the *i*th layer, and $I_i = \frac{Bt_i^3}{12}$ is the second moment of inertia of the *i*th layer of the beam about its center.

Notice that $EI \propto Bh^3$, so that $\mathbf{DA}^{-1} \propto \left(\frac{1}{Bh^3}h\right)(Bh) = \frac{1}{h}$, and so, like bimorph curvature, this:

1. is proportional to the induced strain, and
2. is inversely proportional to the thickness.

This is programmed using MATLAB and compared with the Timoshenko formula for curvature in Appendix C.

Section 2.2.3 Flexural Rigidity

The quantity *EI*, the Young's modulus multiplied by second moment of area, is called the "flexural rigidity" and is a very important parameter for beam bending, being analogous to the spring constant for extension or compression. For a uniform material, the Young's modulus is a constant and we can find the second moment of area is given by $I \equiv \iint_A y^2 dxdy$, where *y* is the distance from the neutral axis of the bending and the integral is taken over a cross section of the material, giving $I = \frac{1}{12} Bh^3$. For a hybrid cantilever with multiple layers having different elastic properties, the formula is more complicated, but we can still find an effective *EI* which describes how the beam behaves.

In any case, the flexural rigidity:

1. has dimensions of $\frac{\mathrm{N}}{\mathrm{m}^2} \times \mathrm{m}^4 = \mathrm{N\text{-}m}^2$,
2. determines the bending moment *M* that a beam produces in response to a changed curvature Δc: $M = -EI \times \Delta c$ (analogue: $F = -k \times \Delta x$), and
3. determines the potential energy stored in a curved beam: $U = \frac{1}{2} EI \left(\Delta c\right)^2 L$ (analogue: $U = \frac{1}{2} k \left(\Delta x\right)^2$).

Section 2.2.3.1 Flexural Rigidity for Multimorphs

Following the work of [29], the flexural rigidity of a multimorph is:

$$EI = \sum \left(E_i I_{0i} + A_i E_i \left(y_i - \bar{Y} \right)^2 \right) \qquad (2.1)$$

where

1. $\bar{Y}$ is the *AE* centroid: $\bar{Y} \equiv \frac{\sum A_i E_i y_i}{\sum A_i E_i}$,
    a. $A_i = Bt_i$ is the cross-sectional area of the i^{th} layer, and
    b. E_i is the [effective] Young's modulus of the i^{th} layer in the longitudinal direction,
2. y_i is the vertical centre of mass of the i^{th} layer, and
3. I_{0i} is the moment of inertia of the *i*th rectangular section about its centroid;

   $I_{0i} = \frac{Bt_i^3}{12}$.

This is programmed in MATLAB and compared with the previous formula in Appendix B. Note that still, $EI \propto Bh^3$.

Section 2.2.4 Beam Displacement due to Curvature

Suppose we have a cantilever spanning internal coordinates $(s, z) \in \left(0..L, \frac{-h}{2}..\frac{h}{2}\right)$.

To elaborate, the *internal coordinates* (s, z) refer to a specific block of matter in the cantilever, while *external coordinates* (x, y) refer to a specific point in space. The point $(s, z) = (0.5\,\text{mm}, 0)$ is always the same point *on the cantilever*, while $(x, y) = (0.5\,\text{mm}, 0)$ is always the same point *in space*, as shown in Figure 2-4.

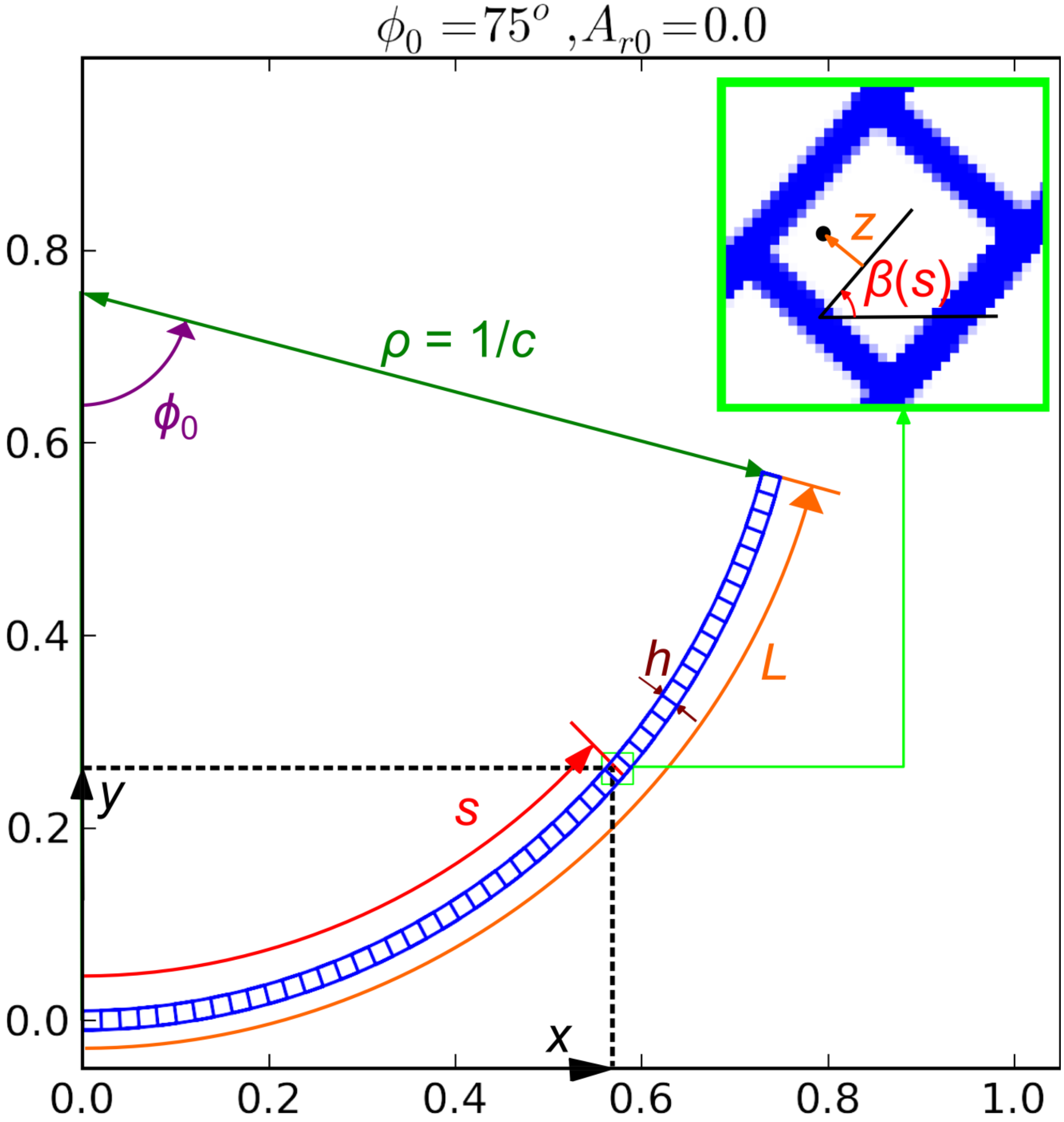


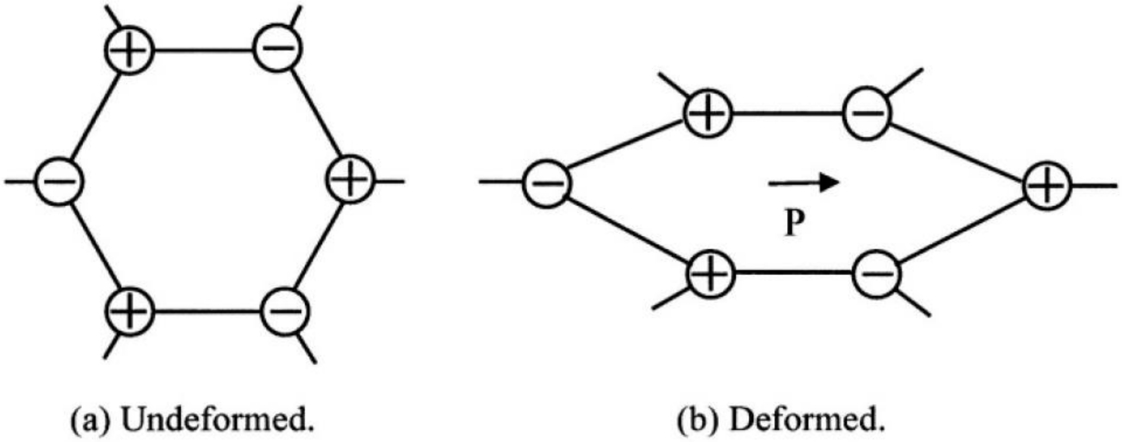
Figure 2-4: Variables for beam displacement; The beam has thickness *h* and length *L*. Location within the beam is determined knowing internal coordinates arclength *s* and perpendicular position *z*. Beam deformation is described by knowing the angle with respect to the horizontal at each position along the arclength, *β*(*s*). Displacement of each point depends on *s*, *β*, and z.

Given the curvature $c \equiv 1/\rho$, what is the displacement $\Delta x(s,z,c)\,\&\,\Delta y(s,z,c)$ of each point on the cantilever, $\begin{bmatrix} x \\ y \end{bmatrix} = \mathbf{r}(s,z,c)$? In words, this problem is "determine the point in space of a point on the cantilever given the cantilever curvature". In the Appendix F, I derive that the solution to this geometric problem is:

$$\mathbf{r} = \begin{bmatrix} s \\ z \end{bmatrix} + \frac{1}{c}\begin{bmatrix} \sin sc - sc \\ 1-\cos sc \end{bmatrix} + z\begin{bmatrix} -\sin sc \\ \cos sc \end{bmatrix}$$

Section 2.2.5 Piezoelectricity and Hook's Law:

Piezoelectric materials have their charge displacement and stress fields coupled together, so that an electric field applied across the material leads to a stress and/or strain, and vice versa. This happens whenever the crystal is made up of atoms with different charges and is sufficiently asymmetric that applied stresses can cause a charge displacement, as in Figure 2-5.



(a) Undeformed.

(b) Deformed.

Figure 1.2-3. Origin of the direct piezoelectric effect.

Figure 2-5 Crystal origin of the piezoelectric effect, from page 11 of [30]

Piezoelectricity is modelled by the piezoelectric constitutive relations. These couple the linear mechanical behaviour (Hooke's law) and electrical behaviour of the material via the piezoelectric coupling coefficient. In stress-charge form:

$$T_{ij} = c_{ijkl} S_{kl} - e_{kij} E_k,$$
$$D_i = e_{ijk} S_{jk} + \varepsilon_{ij} E_j$$

and in strain-charge form:

$$S_{ij} = s^E_{ijkl} T_{kl} + d_{kij} E_k$$
$$D_i = d_{ikl} T_{kl} + \varepsilon^T_{ik} E_k$$

where subscripts denote directions; e.g., T_{11} is linear stress in the *x*-direction, while T_{12} is the *xy*-shear.

Section 2.2.5.1 Reduced Matrix Notation

Since the 2nd order stress and strain tensors are symmetric, we can reduce their 9 matrix components into a 6-by-1 vector with the following convention:

$$S_1 = S_{11}, \quad S_2 = S_{22}, \quad S_3 = S_{33},$$
$$S_4 = 2S_{23}, \quad S_5 = 2S_{13}, \quad S_6 = 2S_{12}$$

This subsequently reduces the 3rd & 4th order coupling coefficient tensors to 3-by-6 and 6-by-6 matrices:

$$\begin{matrix} T_p = c^E_{pq} S_q - e_{kp} E_k \\ D_i = e_{iq} S_q + \varepsilon^S_{ik} E_k \end{matrix}, \text{ or } \begin{matrix} \{T\} = \left[c^E\right]\{S\} - [e]^T\{E\} \\ \{D\} = [e]\{S\} + \left[\varepsilon^S\right]\{E\} \end{matrix}$$

Written out in full,

$$\begin{bmatrix} T_1 \\ T_2 \\ T_3 \\ T_4 \\ T_5 \\ T_6 \end{bmatrix} = \begin{bmatrix} c^E_{11} & c^E_{12} & c^E_{13} & c^E_{14} & c^E_{15} & c^E_{16} \\ c^E_{21} & c^E_{22} & c^E_{23} & c^E_{24} & c^E_{25} & c^E_{26} \\ c^E_{31} & c^E_{32} & c^E_{33} & c^E_{34} & c^E_{35} & c^E_{36} \\ c^E_{41} & c^E_{42} & c^E_{43} & c^E_{44} & c^E_{45} & c^E_{46} \\ c^E_{51} & c^E_{52} & c^E_{53} & c^E_{54} & c^E_{55} & c^E_{56} \\ c^E_{61} & c^E_{62} & c^E_{63} & c^E_{64} & c^E_{65} & c^E_{66} \end{bmatrix} \begin{bmatrix} S_1 \\ S_2 \\ S_3 \\ S_4 \\ S_5 \\ S_6 \end{bmatrix} - \begin{bmatrix} e_{11} & e_{21} & e_{31} \\ e_{12} & e_{22} & e_{32} \\ e_{13} & e_{23} & e_{33} \\ e_{14} & e_{24} & e_{34} \\ e_{15} & e_{25} & e_{35} \\ e_{16} & e_{26} & e_{36} \end{bmatrix} \begin{bmatrix} E_1 \\ E_2 \\ E_3 \end{bmatrix}$$

and

$$\begin{bmatrix} D_1 \\ D_2 \\ D_3 \end{bmatrix} = \begin{bmatrix} e_{11} & e_{12} & e_{13} & e_{14} & e_{15} & e_{16} \\ e_{21} & e_{22} & e_{23} & e_{24} & e_{25} & e_{26} \\ e_{31} & e_{32} & e_{33} & e_{34} & e_{35} & e_{36} \end{bmatrix} \begin{bmatrix} S_1 \\ S_2 \\ S_3 \\ S_4 \\ S_5 \\ S_6 \end{bmatrix} + \begin{bmatrix} \varepsilon_{11}^S & \varepsilon_{12}^S & \varepsilon_{13}^S \\ \varepsilon_{21}^S & \varepsilon_{22}^S & \varepsilon_{23}^S \\ \varepsilon_{31}^S & \varepsilon_{32}^S & \varepsilon_{33}^S \end{bmatrix} \begin{bmatrix} E_1 \\ E_2 \\ E_3 \end{bmatrix}$$

The coupling coefficients in stress-charge and strain-charge form are related by

$$\left[s^E\right] = \left[c^E\right]^{-1}$$

$$[d]^T = \left[c^E\right]^{-1} [e]^T$$

$$[d] = [e]\left[c^E\right]^{-1}$$

$$\left[\varepsilon^T\right] = \left[c^E\right]^{-1} [e]^T + \left[\varepsilon^S\right] = [d]^T + \left[\varepsilon^S\right]$$

The crystal class determines the value of the coupling coefficient tensor entries. Quartz, for instance, has this set of matrix forms:

$$\begin{bmatrix} c_{11} & c_{12} & c_{13} & c_{14} & 0 & 0 \\ c_{12} & c_{11} & c_{13} & -c_{14} & 0 & 0 \\ c_{13} & c_{13} & c_{33} & 0 & 0 & 0 \\ c_{14} & -c_{14} & 0 & c_{44} & 0 & 0 \\ 0 & 0 & 0 & 0 & c_{44} & c_{14} \\ 0 & 0 & 0 & 0 & c_{14} & c_{66} \end{bmatrix}$$

$$\begin{bmatrix} e_{11} & -e_{11} & 0 & e_{14} & 0 & 0 \\ 0 & 0 & 0 & 0 & -e_{14} & -e_{11} \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \begin{bmatrix} \varepsilon_{11} & 0 & 0 \\ 0 & \varepsilon_{22} & 0 \\ 0 & 0 & \varepsilon_{33} \end{bmatrix}$$

We can obtain the specific values of coefficients from COMSOL’s MEMS library:

$$[d]=\begin{bmatrix} -2.3 & +2.3 & 0 & -0.67 & 0 & 0 \\ 0 & 0 & 0 & 0 & +0.67 & +4.6 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}\frac{\text{pN}}{\text{C}}$$

$$[s]=\begin{bmatrix} 12.77 & -1.79 & -1.22 & -4.5 & 0 & 0 \\ -1.79 & 12.77 & -1.22 & 4.5 & 0 & 0 \\ -1.22 & -1.22 & 9.6 & 0 & 0 & 0 \\ -4.5 & 4.5 & 0 & 20.04 & 0 & 0 \\ 0 & 0 & 0 & 0 & 20.04 & -9 \\ 0 & 0 & 0 & 0 & -9 & 29.1 \end{bmatrix}\frac{\text{pm}}{\text{m-Pa}}$$

$$[\varepsilon_r]=\begin{bmatrix} 4.52 & 0 & 0 \\ 0 & 4.52 & 0 \\ 0 & 0 & 4.68 \end{bmatrix}$$

Section 2.2.6 Stress and Strain: Definitions and Relations

Strain is the change in length per unit length: $\varepsilon=\frac{L-L_0}{L_0}$. Axial *stress* is the axial force per unit area: $\sigma=\frac{F}{A}$. The stress and strain in a multimorph vary across the layers and with bending. If we assume that plane sections remain plane (no sliding of layers), then the stress and strain within a given layer are uniform in the absence of bending, linear along the thickness direction with discontinuities only at the layer junctions, and constant along the longitudinal direction.

For a free object in the linear region acted on by an external stress, the external stress produces a proportional strain in the linear region $\varepsilon=\frac{\sigma}{E}$ (with appropriate additions for extra dimensions, depending on boundary conditions).

Composite multimorph strain analysis is more complicated, especially in the presence of piezoelectric, thermal, and polymerization effects. For convenience, define three distinct types of strain:

1. Actual strain, ε, is the change in length per unit length relative to some reference length.
2. Free strain, ε_{free}, is the strain that *would* happen if the layer were free of stresses from other layers. This is due to thermal strain, piezoelectric strain, etc.
3. Mechanical strain, $\varepsilon_{mech} = \sigma / E$, is the forced strain component. This is due to a physical push or pull from other layers.

The first of these, actual strain ε, is measured relative to some reference point, usually defined as when the multimorph is assembled. It is a combination of *mechanical strain* (the strain provided by stress) and the *free strain* (the strain, relative to the reference point, that would occur in each layer if it were liberated from the multimorph):

$$\varepsilon = \varepsilon_{mech} + \varepsilon_{free}$$

These definitions of strains are unique to this work, and prove very helpful aids in discussing multimorph bending. The next three sections provide further detail on these three strain definitions.

Section 2.2.6.1 Total Strain

As long as the multimorph stays intact, then the total longitudinal strain of any point relative to when the multimorph was assembled is linear in *z*, the distance from the bottom of the multimorph, $\varepsilon = \varepsilon_0 + \varepsilon_{bending}$, where ε_0 is the arclength change of the bottom, and $\varepsilon_{bending} = -z\dfrac{\phi}{L}$ is the change in the strain due to bending. For positive-ϕ bending (upward bending), higher layers (higher *z*) experience more compression than lower layers.

Section 2.2.6.2 Free Strain

Free strain is the strain that would occur if the layer were released.

$$\varepsilon_{free} = \varepsilon_{thermal} + \varepsilon_{polymerization} + \varepsilon_{deposition} + \varepsilon_{piezo}$$

$$\varepsilon_{free} = \varepsilon_{free0} + \varepsilon_{piezo}$$

$$\varepsilon_{thermal} = \alpha \Delta T$$

$\varepsilon_{polymerization}$ = some fixed value

$\varepsilon_{piezo} = E_3 d_{31}$, (or appropriate)

Since the piezoelectric strain is the only time-varying strain (for piezoelectrically-actuated wings), break the free strain into ε_{free0} and ε_{piezo}.

Negative values for free strain parameters mean the material would initially shrink if freed from the multimorph.

Section 2.2.6.3 Mechanical Strain

$\varepsilon_{mech} = \dfrac{\sigma}{E}$ is constant within the layer until bending happens, after which it is a linear function of *z*:

$$\varepsilon_{mech} = \varepsilon - \varepsilon_{free}$$

$$\varepsilon_{mech} = \varepsilon_0 + \varepsilon_{bending} - \varepsilon_{free}$$

$$\frac{\sigma}{E} = \varepsilon_0 - \frac{z}{\rho} - \varepsilon_{free}$$

Hence, we can proceed like DeVoe et al. in [28] and break up the mechanical strain into an axial and bending part:

$$\varepsilon_{mech} = \varepsilon_{axial} + \varepsilon_{bending}$$

$$\frac{\sigma_i(z)}{E_i} = \frac{F_i}{A_i E_i} - \frac{z_i}{\rho}$$

where z_i is the coordinate of the point relative to the centre of the *i*th layer, A_i is the area of the *i*th layer, E_i is its Young's modulus, F_i is the total axial force experienced by the layer, *r* is the radius of curvature of the multimorph (assumed not to vary significantly across layers; $\rho >> h$), and σ_i is the stress in the *i*th layer.

Section 2.2.6.4 Curvature from Free Strains: C$_0$ and C$_{piezo}$

As mentioned above, its useful to break the free strain into built-in strain and voltage dependent strain for each layer:

$$\varepsilon_{free} = \varepsilon_{thermal} + \varepsilon_{polymerization} + \varepsilon_{piezo}$$

$$\equiv \varepsilon_{free0} + \frac{d_{31}V}{t}$$

These lead to $\mathbf{C}_0 \,\&\, \mathbf{C}_{piezo}\left(V\right)$ and, through left-multiplication by $\dfrac{1}{1-\frac{1}{2}\mathbf{DA}^{-1}\mathbf{B}}\mathbf{DA}^{-1}$, to c_0 & c_{Piezo}, and hence $\phi_0 \,\&\, \Phi_{DC}\left(V\right)$, respectively. Φ_{DC} and $\mathbf{C}_{piezo}$ are both directly proportional to *V*.

Section 2.2.6.5 Stress at the Bottom and Top of Each Layer

Stress is related to mechanical strain: $\sigma = E\varepsilon_m$

For a material that bends up ($\rho > 0$), the peak compressive strains are located at the top of each layer while the peak tensile strains are located at the bottom of each layer.

$$\varepsilon_{mech} = \varepsilon_{axial} + \varepsilon_{bending}$$

$$\frac{\sigma_i\left(z_{bot}\right)}{E_i} = \frac{F_i}{A_iE_i} - \frac{\left(-t_i/2\right)}{\rho}$$

$$\frac{\sigma_i\left(z_{top}\right)}{E_i} = \frac{F_i}{A_iE_i} - \frac{\left(+t_i/2\right)}{\rho}$$

This is calculated in the Matlab program found in Appendix E.

Section 2.2.6.6 Modifications to *E* & d_{31} for Cantilevers to Account for 3D Effects

DeVoe et al. in [28] mentions that to account for 3D effects, one must replace the Young's modulus and piezoelectric coupling coefficients with effective 3D ones: $E' = E\frac{1}{1-\nu^2}$ and $d_{31}' = d_{31}(1+\nu)$. This results from the plane-strain boundary condition, where the cantilever's boundary conditions prevent it from straining in the *B* or *h* directions. Even then, the effective piezoelectric coupling coefficient equation is only appropriate for the "typical" piezoelectric case where $d_{31} = d_{32}$. For a III-V material with $d_{31} = -d_{32}$, I have found using COMSOL that actually $d_{31}' = d_{31}(1-\nu)$. For x-cut quartz with the crystal *x*-axis in the cantilever's thickness direction and the crystal *y*-axis in the cantilever's length direction, I have found that $d_{21}' = 106\% d_{21}$. Crystal quartz also is sufficiently asymmetric that even the strain condition must be modified, to $E_{\text{effective}}$ = 86.373 GPa. These apply for both resonant frequency and curvature calculations.

These are derived using COMSOL in Appendix H.

Section 2.3 The RS Bending Mode

The small-amplitude resonant mode shape of a flat beam is fairly straight-forward to derive. I repeat such a derivation in Appendix D, resulting in a displacement shape as a function of arclength *s* along a beam of length *L*:

$$\hat{w}(s) = A_1\left[\cosh\frac{\lambda}{L}s - \cos\frac{\lambda}{L}s + \frac{\cos\lambda + \cosh\lambda}{\sin\lambda + \sinh\lambda}\left(\sin\frac{\lambda}{L}s - \sinh\frac{\lambda}{L}s\right)\right]$$

and curvature:

$$\Delta c(s) \equiv \frac{\partial^2 \hat{w}}{\partial s^2} = A_1\frac{\lambda^2}{L^2}\left[\cosh\frac{\lambda}{L}s + \cos\frac{\lambda}{L}s + \frac{\cos\lambda + \cosh\lambda}{\sin\lambda + \sinh\lambda}\left(-\sin\frac{\lambda}{L}s - \sinh\frac{\lambda}{L}s\right)\right]$$

with resonant frequency:

$$\omega_{resStraightBeam} = \lambda^2 \sqrt{\frac{EI}{\rho_m Bh^3}} \frac{h}{L^2}$$

where *B* is the beam's width, *h* is its thickness, ρ_m is its density, *EI* is flexural rigidity ($EI \propto Bh^3$), and λ are the roots of $\cos\lambda \cosh\lambda + 1 = 0$. For the first mode, $\lambda \approx 1.875$. Often, derivations will express this in terms of a pure transverse displacement from resting position. This is fine for small amplitudes, but breaks down for displacements larger than a few percent of the length. Instead, using a computer to iteratively calculate the global beam displacement due to this locally-induced curvature leads to a more accurate picture of beam bending, as shown below in Figure 2-6.

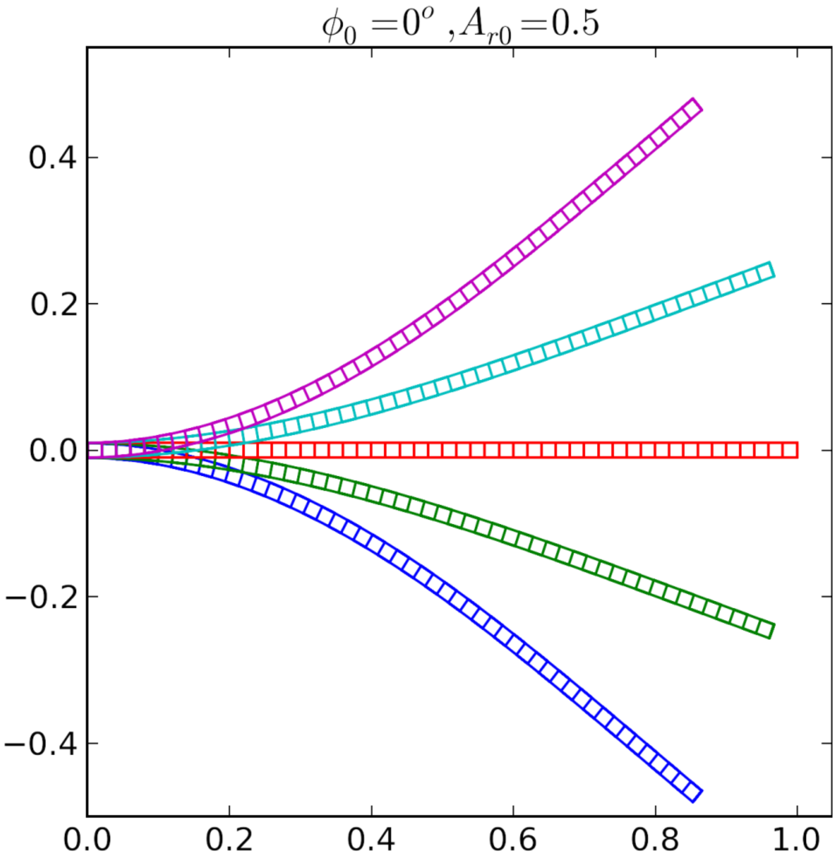


Figure 2-6: Large-amplitude fundamental resonant mode shape for a straight beam

Unfortunately, no concise description for the resonant shape of a naturally curved beam (such as the one in Figure 2-7) exists in literature. Many authors have attempted the problem (e.g., [31]–[34]), but to date it remains a topic for finite element methods. This is unfortunate, since individual experimenters and designers are left with either diving into applied math or redoing and reinterpreting complicated FEM simulations undoubtedly done by others. Further, simulation results for the resonant shape of beams at large amplitudes are often misleading if available at all. For thin cantilevers, the strains stay small and not very different from a straight beam, so it is not unreasonable to expect a solution similar to that of a straight beam. By asserting this, we arrive at what I call the "RS" bending mode (RS referring to "Resonant Shape"):

A thin resonating curved beam should have the same extra bending angle per arc length $\left(\frac{\partial\beta(s)}{\partial s}\right)$ *as a flat one.*

(β is shown in Figure 2-4: Variables for beam displacement.)

This mode then explicitly specifies the wing position via the central longitudinal strain ε_0 and angle β as a function of arclength *s*. The bending of each piece comes from the pieces before it:

$$\mathbf{r}(s,z,t)=\begin{bmatrix}0\\z\end{bmatrix}+\int_0^s\frac{d\mathbf{r}}{ds}ds$$

$$\mathbf{r}(s,z,t)=\begin{bmatrix}0\\z\end{bmatrix}+\int_0^s\left(1+\varepsilon_0(s,t)-z\frac{\partial\beta(s,t)}{\partial s}\right)\begin{bmatrix}\cos(\beta(s,t))\\\sin(\beta(s,t))\end{bmatrix}ds$$

where $c(s,t)=\frac{\partial\beta(s,t)}{\partial s}$ is the curvature. The functional forms are the same as a flat beam, but with an offset angle function β_0 due to the natural curvature:

$$\beta=\beta_0+\Delta\beta,$$

$$\varepsilon_0=0$$

where

1. $\beta_0 \equiv c_0 s$ is the section angle of a constant-curvature wing with curvature $c_0 \equiv \phi_0 / L$

2. $\Delta\beta \equiv A_r \frac{\lambda}{2}\left[\sinh\frac{\lambda s}{L} + \sin\frac{\lambda s}{L} + \frac{\cos\lambda + \cosh\lambda}{\sin\lambda + \sinh\lambda}\left(\cos\frac{\lambda}{L}s - \cosh\frac{\lambda}{L}s\right)\right]$ is the extra tip angle at each section due to resonance:

   a. Curvature is the derivative of the tip angle: $c \equiv \frac{\partial\beta}{\partial s}$

   b. This $\Delta\beta$ is the curvature for small-amplitude flat cantilevers. That functional form satisfies the cantilever boundary conditions,

   $$\Delta c(s) \equiv \frac{\partial^2 \hat{w}}{\partial s^2} = A_1 \frac{\lambda^2}{L^2}\left[\cosh\frac{\lambda}{L}s + \cos\frac{\lambda}{L}s + \frac{\cos\lambda + \cosh\lambda}{\sin\lambda + \sinh\lambda}\left(-\sin\frac{\lambda}{L}s - \sinh\frac{\lambda}{L}s\right)\right]$$

3. $A_r = A_{r0}\sin\omega t$ is the relative tip displacement (i.e., relative to *L*; A_{r0} =10% means the tip amplitude is 0.1**L*). $A_r = 2\frac{A_1}{L}$.

Substituting and using the chain rule, Appendix G shows that we can rewrite the position as: $\mathbf{r} = \int_0^s \begin{bmatrix} \cos\beta \\ \sin\beta \end{bmatrix} ds + z\begin{bmatrix} -\sin\beta \\ \cos\beta \end{bmatrix}$

Although it is defined in terms of an integral, we can still use this beam position definition to iteratively calculate the new positions of a resonating curved wing. For example, the following Python function "`rRS()`", an excerpt from the program in Appendix J, accomplishes just that:

```
def rRS(phi0,L,Ar = 0, z = 0, ep0ByAr=0.00, numintervals=100):
    '''Returns the wing position array r(s) for a given Ar & z, where s has
numintervals+1 values from 0 to L

    phi0: full curvature angle in deg.  phi0 = c0*L
    L: wing length
    Ar: relative amplitude.  Ar = 2*A1/L --> A1 = Ar*L/2.
    z: offset from the neutral axis to compute
    ep0ByAr: longitudinal strain divided by Ar.
    numintervals: number of s-intervals to take from 0 to L.  (Points =
numintervals+1)
    '''
    import scipy as sp
    from scipy import cos
    from scipy import cosh
    from scipy import sin
    from scipy import sinh

    N = numintervals
    r=sp.empty((N+1,2))
    c0 = phi0*sp.pi/180/L
    Ds = L*1.0/N

    ep0= ep0ByAr*Ar

    s = sp.linspace(Ds,L-Ds, N)
    r[0,:]=(0,z)

    lam=1.8751
    Clam = (cos(lam)+cosh(lam))/(sin(lam)+sinh(lam))
    Lastbeta = 0
    for n,sval in enumerate(s):
        slam = sval*lam/L
        beta = c0*sval +Ar/2.*lam*(sinh(slam)+sin(slam)+Clam*(cos(slam)-
cosh(slam)))
        c = c0 + Ar/2.*lam**2/L*(cosh(slam)+cos(slam)+Clam*(-sin(slam)-
sinh(slam)))
        r[n+1,:]=r[n,:]+(1+ep0-z*c)*sp.array([cos(beta), sin(beta)])*Ds  #No
chain rule for the z-part

    return r
```

Using this function to plot a wing with a length of $L = 1$ and thickness $h = 0.02$ with bending angle $\phi_0 = 90^o$ when resting generates Figure 2-7.

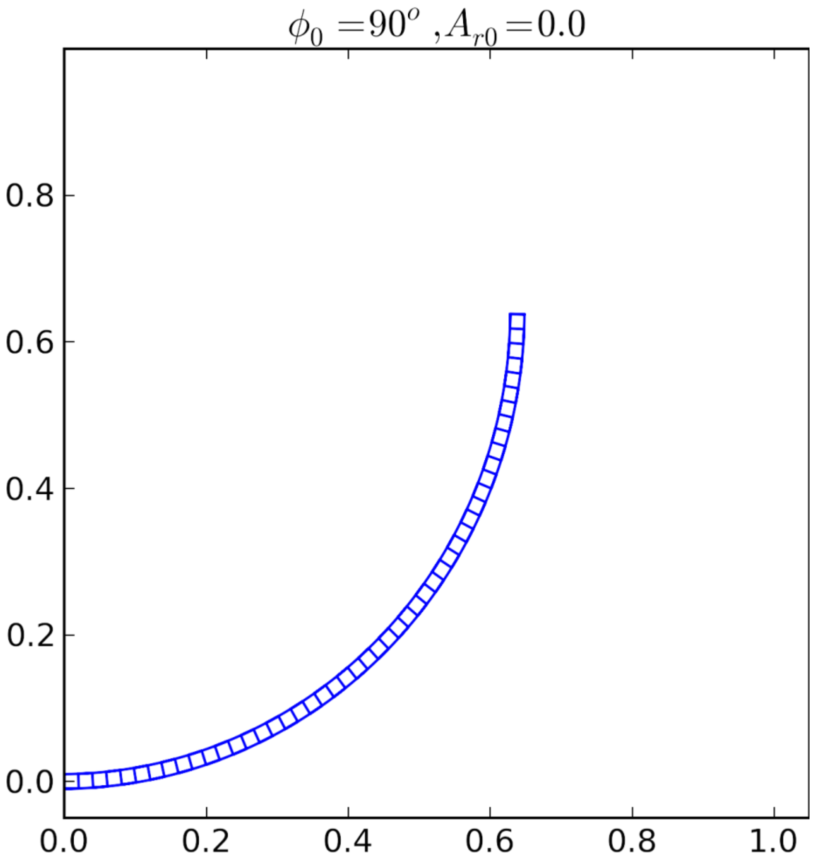


Figure 2-7: A beam with *h/L* = 2% subtending an angle of ϕ_0 = 90°

We can also use `rRS()` to plot this shape at a range of amplitudes, A_r = (-0.5, -0.25, 0, +0.25, +0.5):

```
import matplotlib.pyplot as plt
fig1 = plt.figure(figsize = (6,6))
rRSPlot = fig1.add_subplot(1,1,1)
rRSPlot.set_xlim((0,1))
rRSPlot.set_ylim((-.2,.8))
for Ar in linspace(-.5,.5,5):
    for i,z in enumerate(linspace(-h/2,h/2,5)):
        r =rRS(90, 1,Ar, z=z)
        rRSPlot.plot(r[:,0], r[:,1])
plt.show()
```

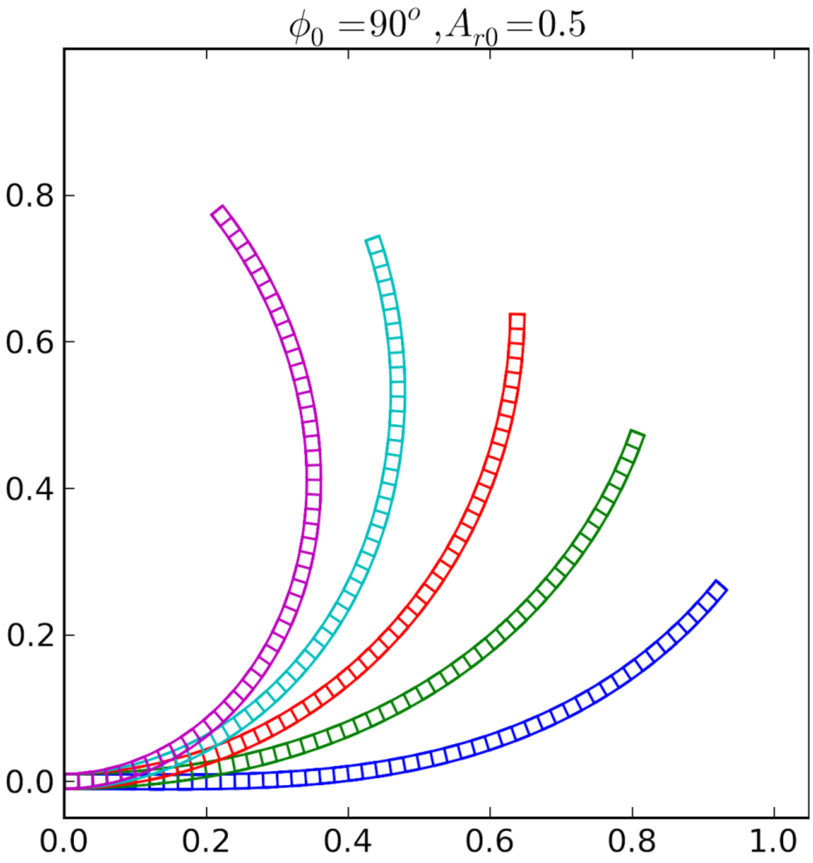


Figure 2-8: A beam with h/L = 2% subtending an angle of ϕ_0 = 90°, resonating with amplitude A_r = 0.5 for the RS mode

Unfortunately analytically deriving the differential equations for resonant curved beams and substituting this mode has proved intractable so far. Nevertheless I have two pieces of evidence that this mode is a reasonable representation of reality:

1. this mode *looks* like it fits well to the actual resonant flapping shape of fabricated curved cantilevers, and
2. this mode correctly predicts the curved beam's resonant frequency modification as a function of bending angle.

Section 2.3.1 Experimental Justification

Although difficult to justify analytically, this bending is a natural fit to the actual wing deformation. Consider the images of resonant curved cantilever wings shown in Figure 2-9 through Figure 2-12. The fabrication of these wings is discussed in Chapter 6. Figure

2-9 shows the wings with no applied voltage while Figure 2-10 shows this pair of wings at resonance. To more easily distinguish the resonating wings from the background, the images are differenced and filtered resulting in Figure 2-11. In Figure 2-10 and Figure 2-11, the RS mode shape is overlaid in red while the DC bending shape of circular arc sections is overlaid in yellow. These images demonstrate that the RS Bending mode is a better fit to the actual resonant shape than purely cylindrical bending shapes. The difference is even more apparent in Figure 2-12, which shows the flapping shape for an even larger amplitude resonance.

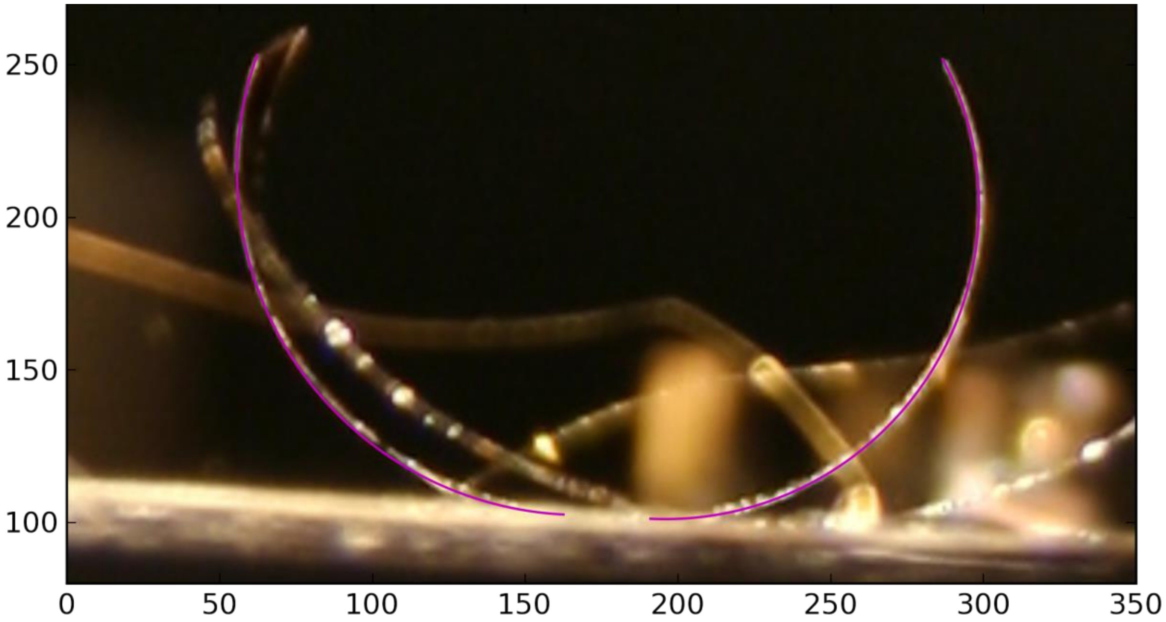


Figure 2-9: Stationary image of cantilever wings with arcs superimposed

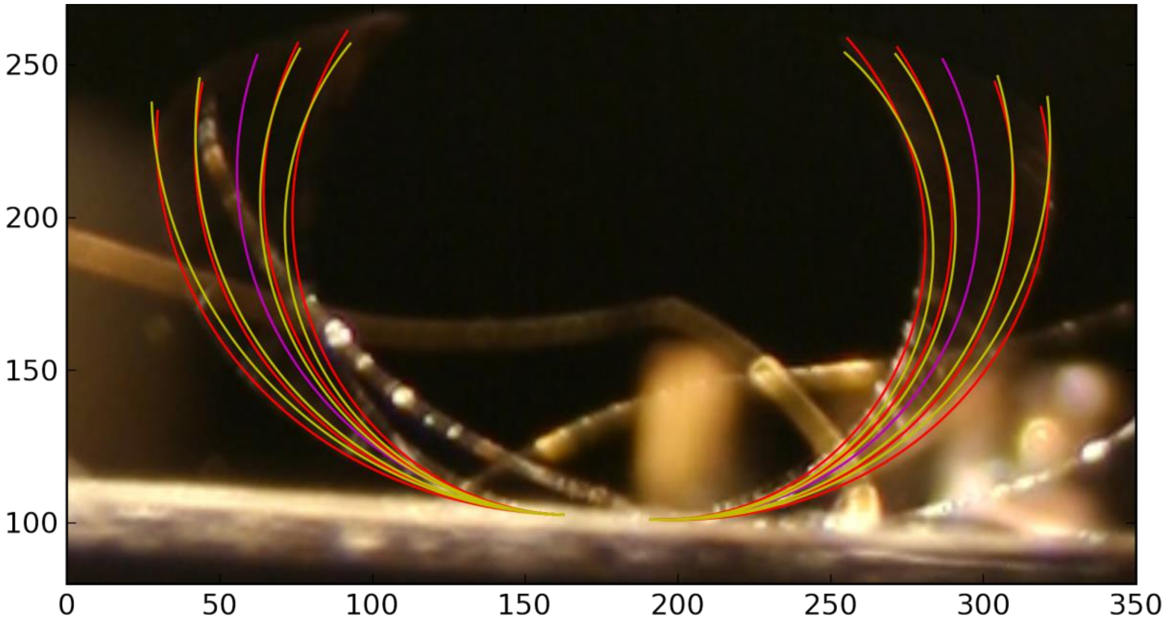


Figure 2-10: Resonating cantilever wings for comparing RS bending (red) to cylindrical arc bending (yellow), which is the non-resonant bending shape. RS bending mode is A_{r0} = 0.18, cylindrical is $\Phi = 21^o$.

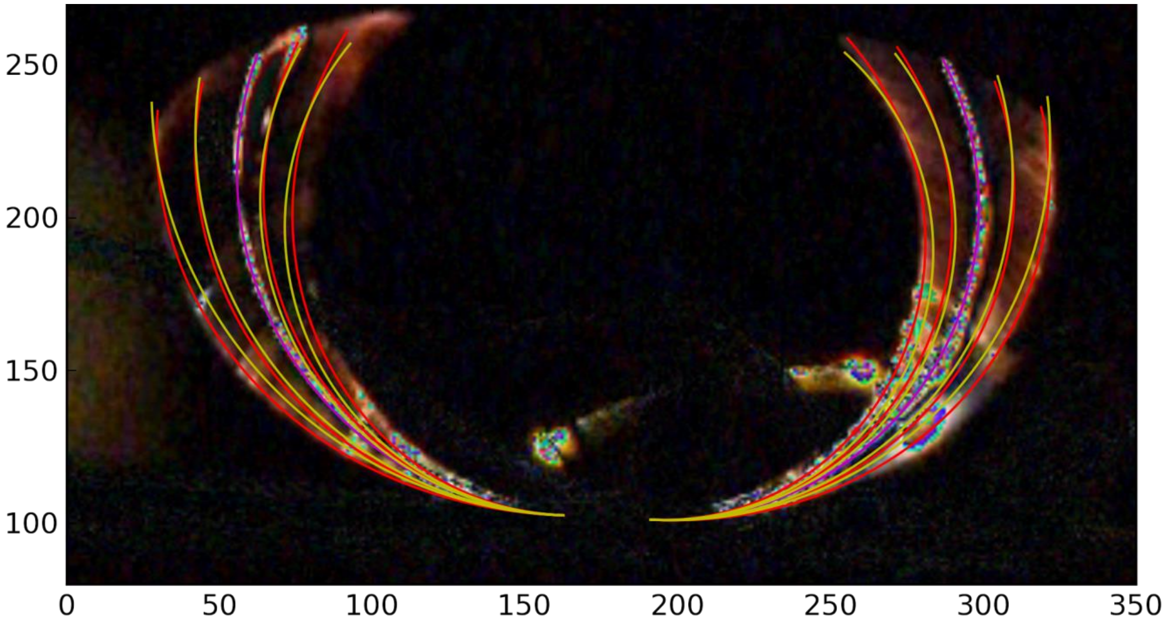


Figure 2-11: Difference image of resonating and stationary cantilever wings comparing RS bending (red) to cylindrical arc bending (yellow), which is the non-resonant bending shape. RS bending mode is A_{r0} = 0.18, cylindrical is $\Phi = 21^o$.

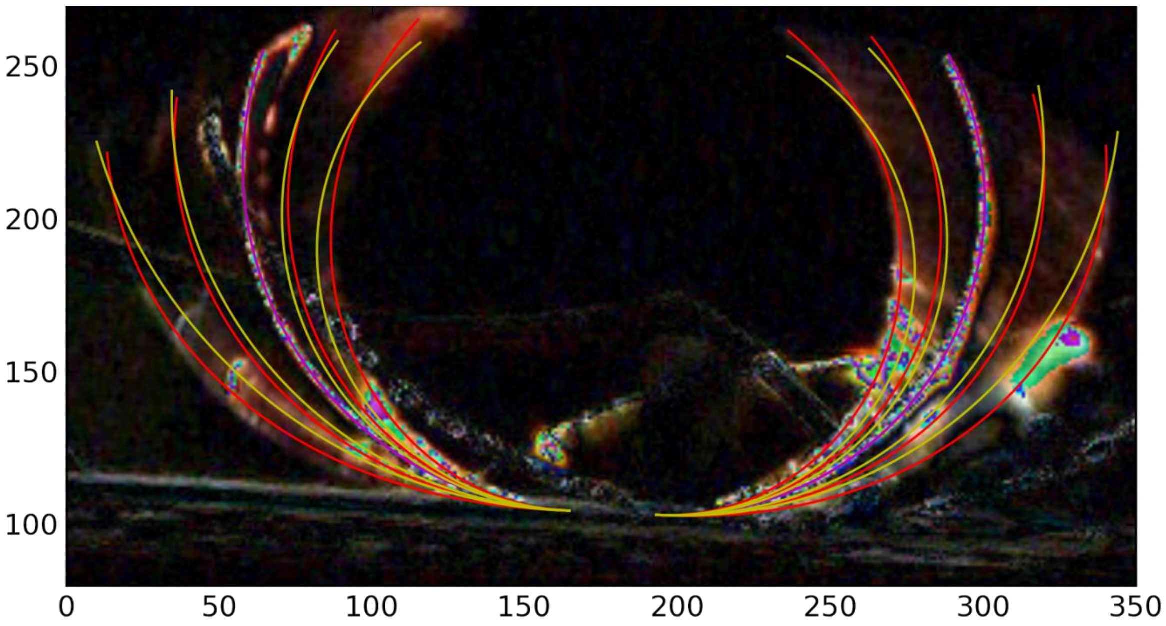


Figure 2-12: RS bending mode arcs fitting to a larger amplitude resonance shape for QSU8_17a. RS bending mode is $A_{r0} = 0.30$, cylindrical is $\Phi = 34^o$.

Section 2.3.2 Resonant Frequency Prediction

Further justification comes from the RS mode's correct prediction of the resonant frequency of cantilevers. It is possible to determine the resonant frequency of a harmonically resonating structure by equating the peak kinetic and potential energies. Appendix I derives that the peak kinetic and potential energy of the RS mode shape are given by

$$K_{\max} = \tfrac{1}{2}\int_0^L v_{\max}^2(s,t)\mu_m ds = \frac{A_{r0}^2\omega^2 L^3}{8 f_{resCor}^2}(\rho_m Bh) \quad (2.2)$$

and

$$\Delta U_{\max} = \tfrac{1}{2} EI\int_0^L \Delta c^2 ds = \frac{1}{2} EI\left(\frac{\lambda^4}{4}\frac{A_{r0}^2}{L}\right) \qquad (2.3)$$

where $\mu_m = hB\rho_m$ is linear mass density, $\lambda \approx 1.8751$ is the first root of $\cos\lambda\cosh\lambda + 1 = 0$, and the resonant frequency curvature-correction factor is:

$$f_{resCor} \approx 1 + 2.0\text{e-}2\phi_0^2 + 2.4\text{e-}4\phi_0^4 \qquad (2.4)$$

Equating these two energies at resonance leads to the resonant frequency equation:

$$\frac{A_{r0}^2 \omega_{res}^2 L^3}{8 f_{resCor}^2}(\rho_m Bh) = \frac{1}{2} EI\left(\frac{\lambda^4}{4}\frac{A_{r0}^2}{L}\right)$$

$$\omega_{res} = \lambda^2 \sqrt{\frac{EI}{\rho_m Bh^3}}\frac{h}{L^2} f_{resCor}$$

(or, $f = \frac{\lambda^2}{2\pi}\sqrt{\frac{EI}{\rho_m Bh^3}}\frac{h}{L^2} f_{resCor}$)

Comparing to the resonant frequency equation for a straight beam,

$\omega_{resStraightBeam} = \lambda^2 \sqrt{\frac{EI}{\rho_m Bh^3}}\frac{h}{L^2}$, the curved beam has a resonant frequency given by the same formula multiplied by the correction factor, $f_{resCor} = 1 + 2.0239\text{e-}2\phi_0^2 + 2.4241\text{e-}4\phi_0^4$.

This correction factor analytically explains numeric results I have obtained using ANSYS Mechanical, and published results by Zhang et al. in [35], as shown in Figure 2-13.

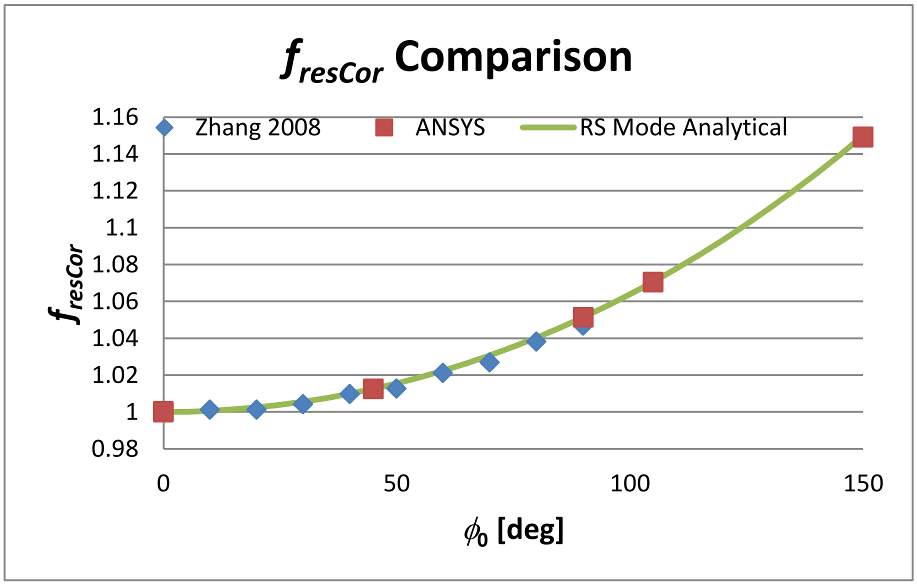


Figure 2-13: Comparing the f_{resCor} with numerical results

Section 2.3.3 Nonlinearities

At sufficiently large amplitudes, resonant systems typically become nonlinear as higher order terms in Taylor expansions are no longer ignorable. Nonlinear systems give rise to nonlinear resonance effects. For example, Landau and Lifshitz in chapters 28 and 29 of [36] suggest that at large amplitudes, the resonant frequency depends on the amplitude: $\omega = \omega_0 + \kappa b^2$ (where *b* is the amplitude).

Another complication is the "foldover" effect: the resonant amplitude at a given frequency can depend on whether that frequency is approached from above or below, as illustrated in Figure 2-14.

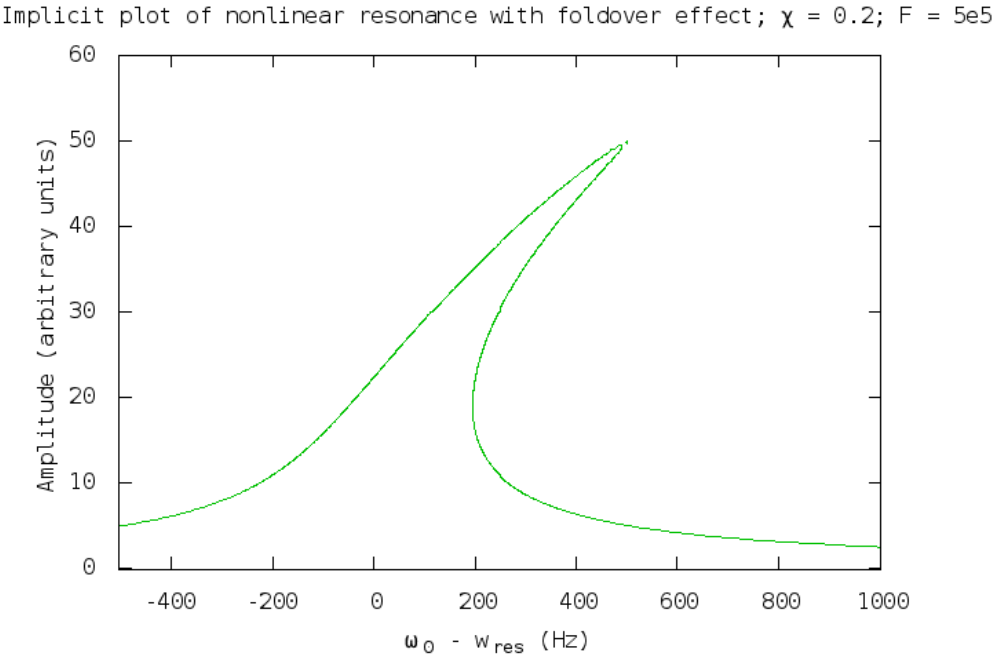


Figure 2-14: The foldover effect of nonlinear resonance leads to different amplitudes at the same frequency and driving force [37]

Chapter 3 CFD Simulations

Even if you know exactly how a flapping wing is moving, calculating the force on it is difficult because the Navier-Stokes equations (the governing differential equations for incompressible fluid mechanics, which are a statement of Newton's second law for continuum mechanics), are nonlinear:

$$\rho\left(\frac{\partial \mathbf{v}}{\partial t}+\mathbf{v}\cdot\nabla\mathbf{v}\right)=-\nabla p+\mu\nabla^2\mathbf{v}+\mathbf{f} \qquad (3.1)$$

where **v** is the local velocity vector of the fluid, ρ is the fluid density, μ is the fluid dynamic viscosity, and **f** contains other body forces, such as gravity [38]. Fortunately, these are still numerically solvable, and computers have recently made solving these problems on a large scale much more tangible using finite element methods.

In fluid mechanic analysis of airplane wings or propeller blades, the problem is greatly simplified by using steady-state flow conditions. However, initial simulations revealed that the instantaneous force on oscillating wings is not equal to the force from extending steady-state flow results. For this reason, it was necessary to complete transient fluid mechanics simulations of flapping wings and average over several cycles (after the flow had stabilized) to determine the net force and average power dissipation of the wings.

To complete these simulations, I used the commercial fluid mechanics software ANSYS CFX (CFX), run on SHARCNET[3]. A tutorial detailing specifically how to set up a representative simulation is given in Appendix K.

In this section, I describe some fluid mechanics basics and then the results of CFD simulations of flapping curved cantilevers.

[3] SHARCNET (www.sharcnet.ca) is a consortium of colleges, universities and research institutes operating a network of high-performance computer clusters across south western, central and northern Ontario.

Section 3.1 Fluid Mechanics Background Information

Section 3.1.1 Physical Constants: Density and the Viscosities

Three relevant constants for fluid mechanics are the fluid density ρ, the dynamic viscosity μ, and the kinematic viscosity ν. The three are related by

$$\nu = \frac{\mu}{\rho}$$

The viscosities are measures of the difficulty a fluid experiences trying to slide past itself.

As explained in Appendix L, dynamic viscosity μ for a gas is largely pressure-independent below 30 atm, and since density is inversely proportional to pressure by the ideal gas law, kinematic viscosity is directly proportional to it.

For air at 20 ^{o}C, ρ_{air} = 1.2041kg/m^3 and ν_{air} = 1.46e-5 m^2/s [39].

Section 3.1.2 Similarity of Flows

For incompressible lossless flow, the Navier-Stokes equations by themselves completely govern the velocity field of the fluid. Since these equations are time consuming to numerically solve, it is useful to get as much information as possible out of every solution. One way to do this is by "nondimensionalizing" the equations, as in [38].

To nondimensionalize the Navier-Stokes equations, define dimensionless quantities q^* that are proportional to the actual quantities q through appropriate scale quantities, q_0. For example, define the dimensionless coordinates (x^*, y^*, z^*) as the actual coordinates divided by the length scale, L: $(x^*, y^*, z^*) \equiv \left(\frac{x}{L}, \frac{y}{L}, \frac{z}{L}\right)$. Similarly, define dimensionless:

1. velocity as $\mathbf{v}^* \equiv \mathbf{v} / v_0$, where v_0 is the velocity scale,

2. time as $t^* \equiv t / t_0$, where $t_0 \equiv \dfrac{L}{v_0}$ is the time scale,
3. del operator as $\nabla^* \equiv L\nabla$, and
4. pressure as $p^* \equiv p / p_0$, where $p_0 \equiv \rho v_0^2$ is the pressure scale (this definition requires incompressible flows so that the density is a constant).

Substituting into the Navier-Stokes equations,

$$\rho\left(\frac{\partial\left(v_0\mathbf{v}^*\right)}{\partial\left(\dfrac{L}{v_0}t^*\right)}+\left(v_0\mathbf{v}^*\right)\cdot\left(\frac{1}{L}\nabla^*\right)\left(v_0\mathbf{v}^*\right)\right)=-\left(\frac{1}{L}\nabla^*\right)\left(p_0 p^*\right)+\mu\left(\frac{1}{L}\nabla^*\right)^2\left(v_0\mathbf{v}^*\right)+\mathbf{f}$$

$$\rho\left(\frac{v_0^2}{L}\frac{\partial\mathbf{v}^*}{\partial t^*}+\frac{v_0^2}{L}\mathbf{v}^*\cdot\nabla^*\mathbf{v}^*\right)=-\frac{\rho v_0^2}{L}\left(\nabla^* p^*\right)+\frac{v_0}{L^2}\mu\left(\nabla^*\right)^2\left(\mathbf{v}^*\right)+\mathbf{f}$$

$$\frac{\partial\mathbf{v}^*}{\partial t^*}+\mathbf{v}^*\cdot\nabla^*\mathbf{v}^*=-\nabla^* p^*+\frac{\mu}{\rho v_0 L}\nabla^{*2}\,\mathbf{v}^*+\frac{L}{\rho v_0^2}\mathbf{f}$$

and if the only body force is gravity, $\mathbf{f}=-\rho g\hat{\mathbf{z}}$, then

$$\frac{\partial\mathbf{v}^*}{\partial t^*}+\mathbf{v}^*\cdot\nabla^*\mathbf{v}^*=-\nabla^* p^*+\frac{\mu}{\rho v_0 L}\nabla^{*2}\,\mathbf{v}^*-\frac{Lg}{v_0^2}\hat{\mathbf{z}}.$$

The result is that we can use a Navier-Stokes solution for one flow situation as the solution for any other flow situation (in terms of relative parameters, that is) which is "similar" to the first one, where two flows are "similar" if they have the same Re, Fr, and C_p where:

1. the Reynolds number is $\mathrm{Re} \equiv \rho\dfrac{v_0 L}{\mu}$, the ratio of inertial to viscous forces,
2. the Froude number is $\mathrm{Fr} \equiv \dfrac{v_0^2}{gL}$, an expression of the relative strength of gravity on the flow scale, and

3. the pressure coefficient C$_p$ = $\dfrac{p(upstream)-p(downstream)}{\frac{1}{2}\rho v_0^2}$, the ratio of local pressure drop to kinetic energy per volume.

Perfect frictionless flow corresponds to C$_p$ = 2; this is assumed for all of my flow situations, and is equivalent to stating that the energy lost to heat is negligible.

Section 3.1.3 The Power of Drag Coefficients: Similarity of Flows

One way to apply similarity of flows is by drag coefficients. The drag force on a body in a fluid is given by $F_D = \frac{1}{2}\rho C_D A v^2$, where ρ is the fluid density, A is the cross sectional area of the body, v is the velocity of the fluid flow past the body, and C_D is the "drag coefficient", which accounts for streamlining.

This is particularly useful because the drag coefficient is the same for two similar flow situations. To see this, define the drag force on a body in terms of the dimensionless force $F^* \equiv F / F_0$,where the force scale is $F_0 \equiv \rho v_0^2 L^2$. The dimensionless force is then $F^* = \dfrac{\frac{1}{2}\rho C_D A v^2}{\rho v_0^2 L^2} = \frac{1}{2} C_D v^{*2} A^*$ (where $A^* \equiv A / L^2$ is the dimensionless version of the body's cross sectional area), which means that the drag coefficient depends only on the flow's dimensionless scales: $C_D = \dfrac{2F^*}{v^{*2} A^*}$, and thus two similar flows will have the same drag coefficient.

For example, suppose we use a fluid mechanics simulation to find 640 nN of drag force on a square plate with side length L = 1 mm sitting in a ρ_{air} = 1 kg/m^3 and $\mu = 1.5\text{e-}5$ Pa-s fluid moving with velocity v = 1 m/s. Then we can calculate that the Reynolds number and drag coefficient for this flow are $\text{Re} = \rho\dfrac{vL}{\mu} = 68.5$ and

$C_D = \dfrac{2F_D}{\rho_{air} A v^2} = 1.28$, respectively. From similarity of flow, we know that the flow over a square plate with side length of L = 10 cm in a fluid with ρ_{air} = 0.5 kg/m^3 and $\mu = 1.5\text{e-}5\ \text{Pa-s}$ moving at v = 2 cm/s will be similar to the situation we simulated, and so the drag coefficient will be the same: 1.28. This is sufficient to calculate that the force in this new flow situation is

$$F_D = \tfrac{1}{2}\rho_{air} C_D A v^2 = \tfrac{1}{2}(0.5\text{kg})(1.28)(0.1\text{m})^2(0.02\text{m/s})^2 = 1.28\mu\text{N}.$$

Note that if the fluid density and viscosity are held constant, then the force is also independent of length and velocity at constant Reynolds number, but the drag coefficient scaling is more general because it stays constant even across different fluid density and viscosity.

Section 3.2 Simulations, Inputs and Outputs

CFX simulation inputs varied were the initial bending angle (ϕ_0), the amplitude (A_r) and the Reynolds number (Re). Since the length and gas type were typically fixed, the Reynolds number was varied via the flapping frequency. CFX is only a fluid mechanics solver, and so these simulations do not consider how fluidic forces are modifying the resonant shape of the wing. It is possible to couple these kinds of solvers together, but at time of writing it is prohibitively computationally intensive. Fortunately, since the quality factor is a ratio of stored to dissipated energy, a sufficiently high quality factor indicates the damping is not appreciably affecting the resonant shape, so these CFD-only simulation results can nevertheless be accurate for that high-Q range.

The CFX simulations output a plethora of information, but most importantly they monitor the net force on the wings and their instantaneous power dissipation. CFX commands make it possible to extract this monitor data into a csv (comma separated values) file, which allows processing it using other programs.

From the csv files, we can compute a moving average of the net force over an integer number of cycles. The moving average of force computed this way is the net force of the simulation (technically, since the simulation models ½ of a wing, this force is really $F_{N0.5}$. The net force of a 4-wing robot is $F_{N4} \equiv 8F_{N0.5}$, and the net force per wing is $F_N \equiv 2F_{N0.5}$. This is another advantage of working with drag coefficients rather than forces: they are area-independent). In the same way, averaging the power dissipation over an integer number of flapping cycles determines the average power dissipated into the fluid, $P_{A0.5}$, as well as the power used by 4 wings, $P_{A4} \equiv 8P_{A0.5}$, and the power used by one wing, $P_A \equiv 2P_{A0.5}$. This net force and power combine with the known simulation inputs to determine the drag coefficients as functions of three "shape" parameters: ϕ_0, A_{r0}, and Re. For these wings, I define the Reynolds number as

$$\boxed{\mathrm{Re} \equiv \rho_{air} \frac{v_{rms}\sqrt{A}}{\mu}} \quad (3.2)$$

where ρ_{air} and μ are the air density and dynamic viscosity, $A = LB$ is the wing area, and $v_{rms} \equiv \sqrt{\frac{\int_0^L v_{\max}^2 ds}{L}}$ is the RMS wing velocity for the RS flapping mode, which turns out to be $v_{rms}(\phi_0) = \frac{v_{rms}(0)}{f_{resCor}(\phi_0)} = \frac{A_{r0}\omega L}{2 f_{resCor}(\phi_0)}$. The choice of which velocity and length scale to use for Reynolds number is not critical, but consistency in that choice is.

Section 3.2.1 Drag Coefficients, C_{DN} and C_{DP}

Drag coefficients are convenient because they do not scale with the number of wings, so there is less risk of ambiguity when finding the total force output for a multi-wing robot. The net drag coefficient is defined as:

$$C_{DN} \equiv \frac{2F_N}{\rho A v_{rms}^2}$$

where the wing area is $A = LB = \frac{L^2}{4}$. Similarly, the power drag coefficient C_{DP} is defined as:

$$C_{DP} \equiv \frac{2F_P}{\rho A v_{rms}^2}$$

where F_P is the "power force", defined as

$$F_P \equiv \frac{P_A}{v_{rms}} \qquad (3.3)$$

By substituting the Reynolds number into the average velocity, we can rewrite the drag coefficient expressions:

$v_{rms} = \frac{\mu \mathrm{Re}}{\rho\sqrt{A}}$, and so

$$\boxed{C_{DP} = \rho\frac{2F_P}{\mu^2 \mathrm{Re}^2}} \ (3.4)$$

and similarly for the net drag coefficient,

$$\boxed{C_{DN} = \rho\frac{2F_N}{\mu^2 \mathrm{Re}^2}} \ (3.5)$$

I am not aware of any prior work which extends the drag coefficient concept from steady state fluid mechanics to transient fluid mechanics, or applies it to power scaling. However, CFD simulations of different lengths, gas densities, and viscosities verify that these drag coefficients are functions only of shape and Reynolds number, meaning we can use them to predict fluid mechanics results using similarity of flow. In summary, each CFX simulation is started with a unique pair of shape parameters ϕ_0 and Ar_0, and a unique Reynolds number Re, and leads to the drag coefficients C_{DN} and C_{DP}, which depend only on these shape parameters and Reynolds number.

Section 3.3 Simulation Results

Figure 3-2 through Figure 3-5 plot the net drag coefficient C_{DN}, power drag coefficient C_{DP}, relative drag coefficient C_{DR} and flapping efficiency (defined as net force divided by average absolute force magnitude) as functions of Reynolds number Re, flapping amplitude A_{r0}, and static bending angle ϕ_0. Each complete plot is interpolated from approximately 200 individual simulations spaced by factors of 2 in Reynolds number and steps of 0.05 in A_{r0}, shown without interpolation in Figure 3-1.

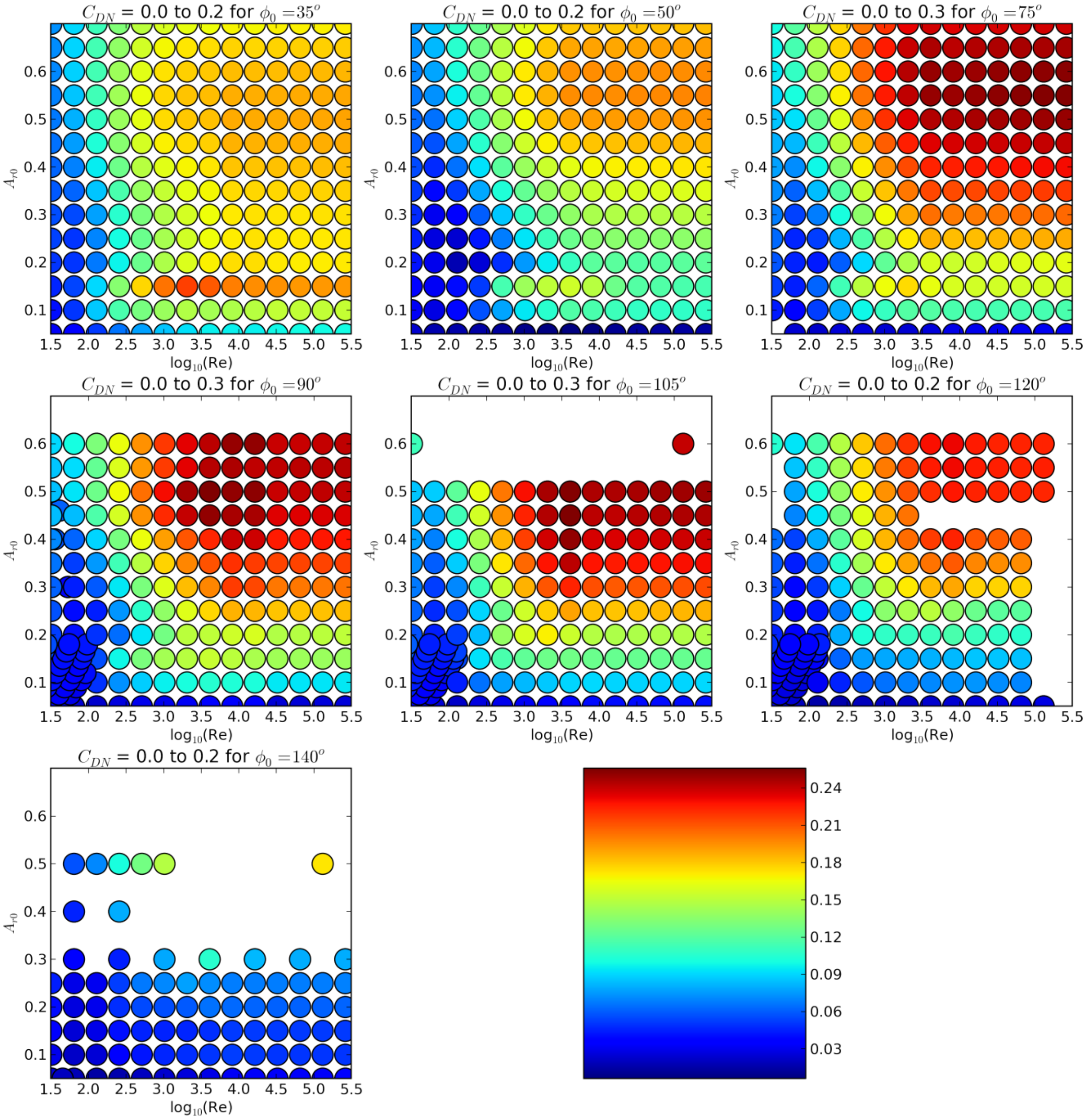


Figure 3-1: Raw data points used to generate contour maps for net drag coefficient (as of 24 Oct 2013, simulations are in progress to fill the holes on this grid)

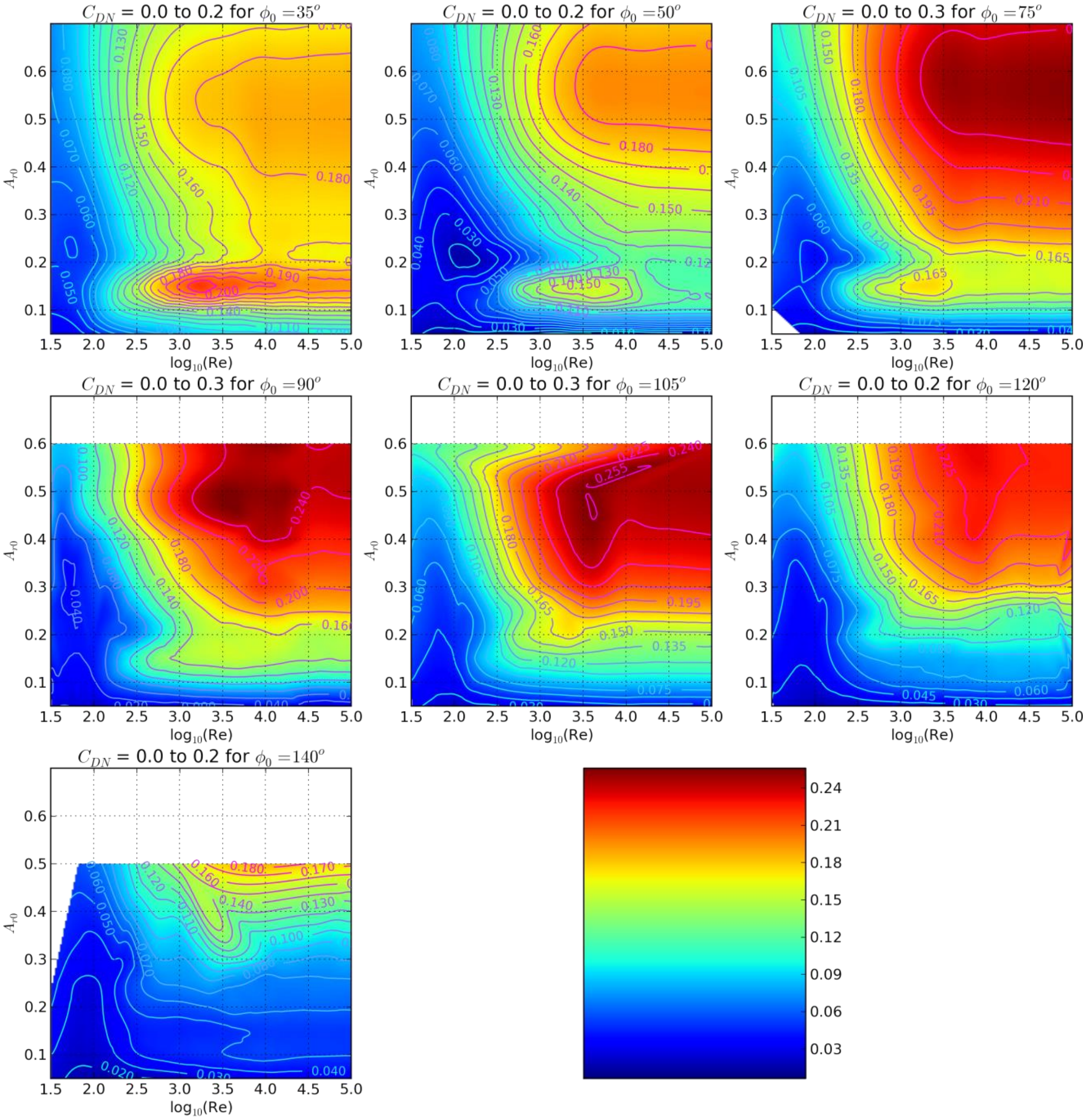


Figure 3-2: Net drag coefficient as a function of Reynolds number and flapping shape

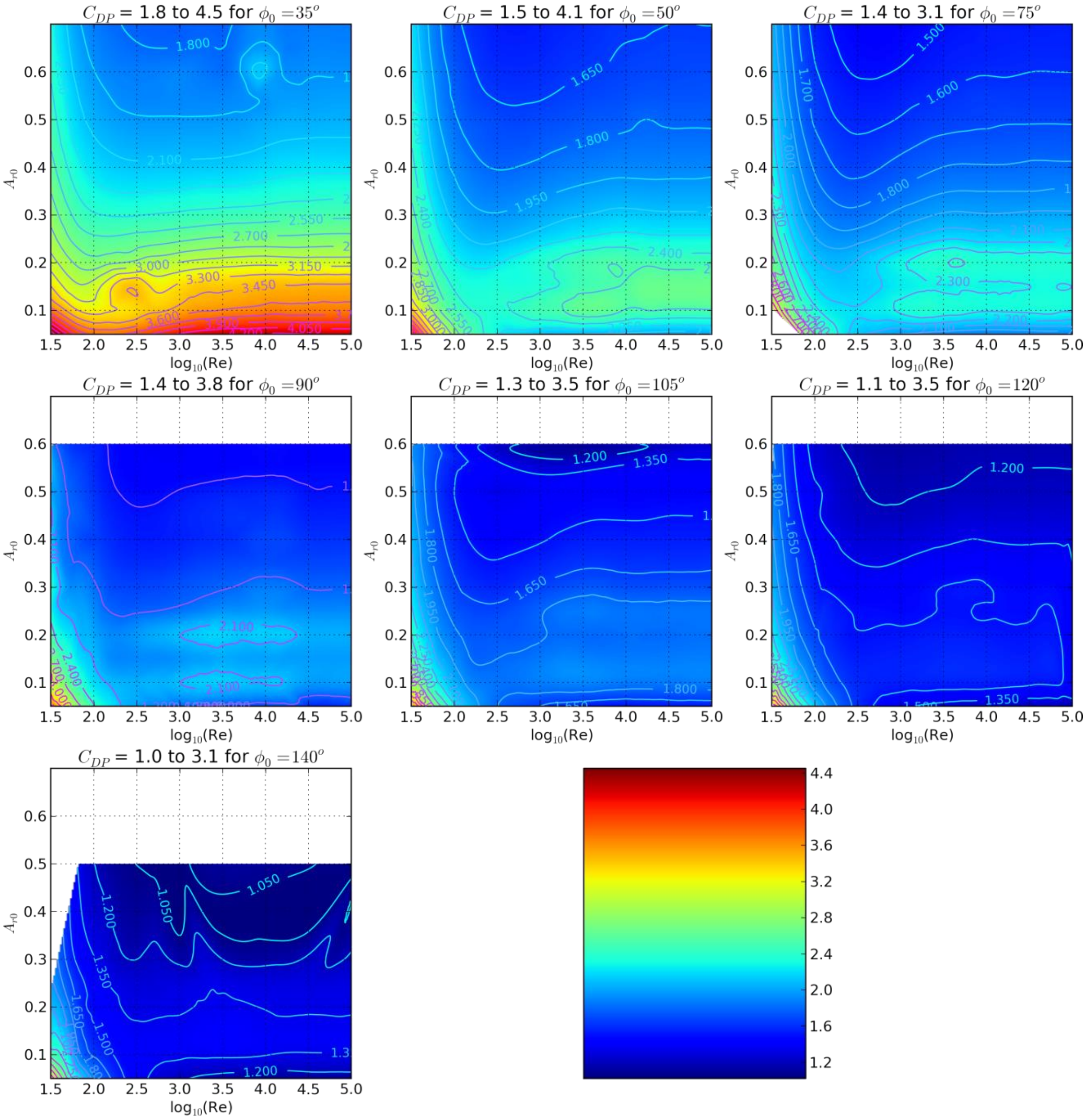


Figure 3-3 Power drag coefficient as a function of Reynolds number and flapping shape

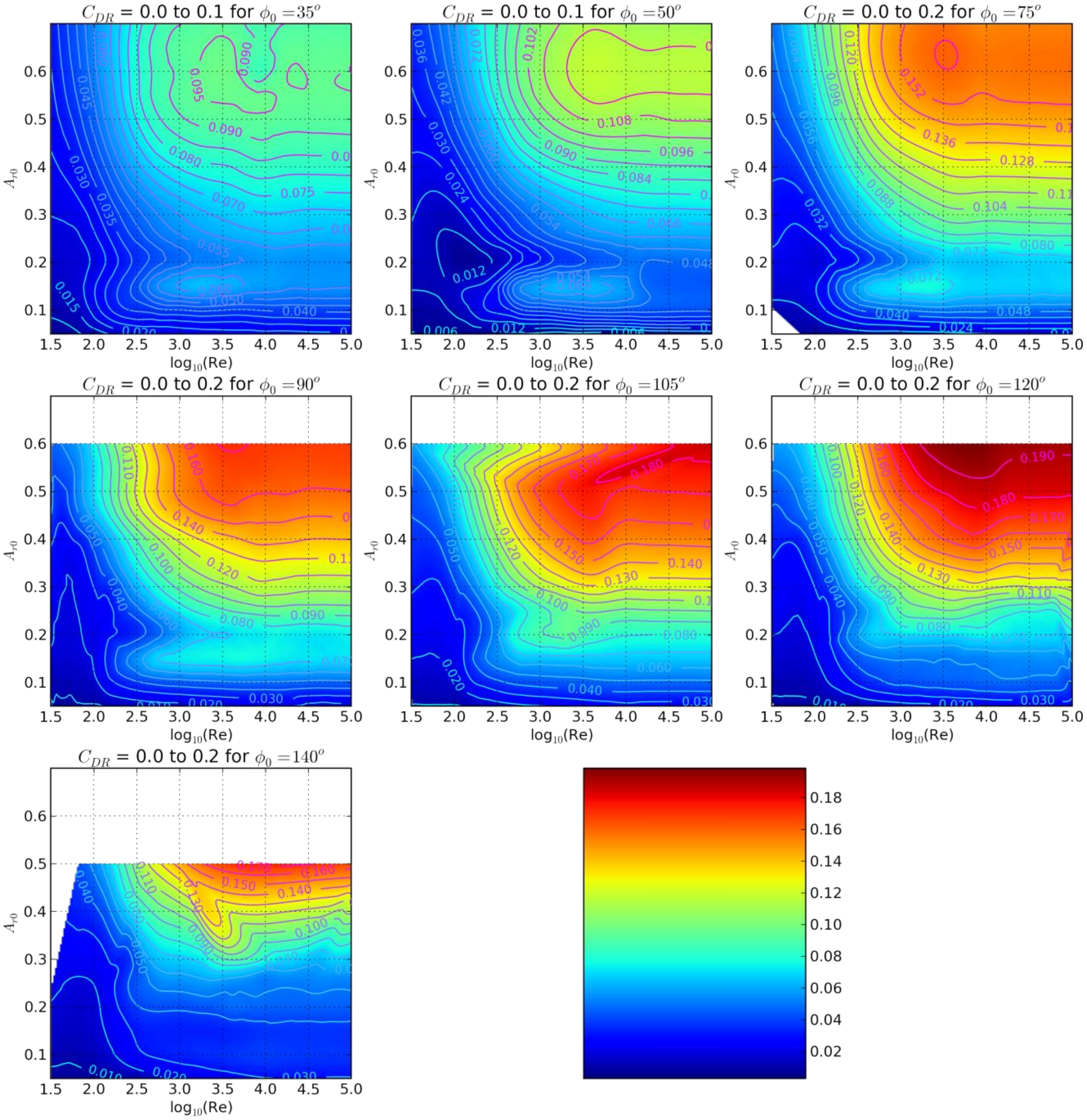


Figure 3-4: Relative drag coefficient as a function of Reynolds number and flapping shape

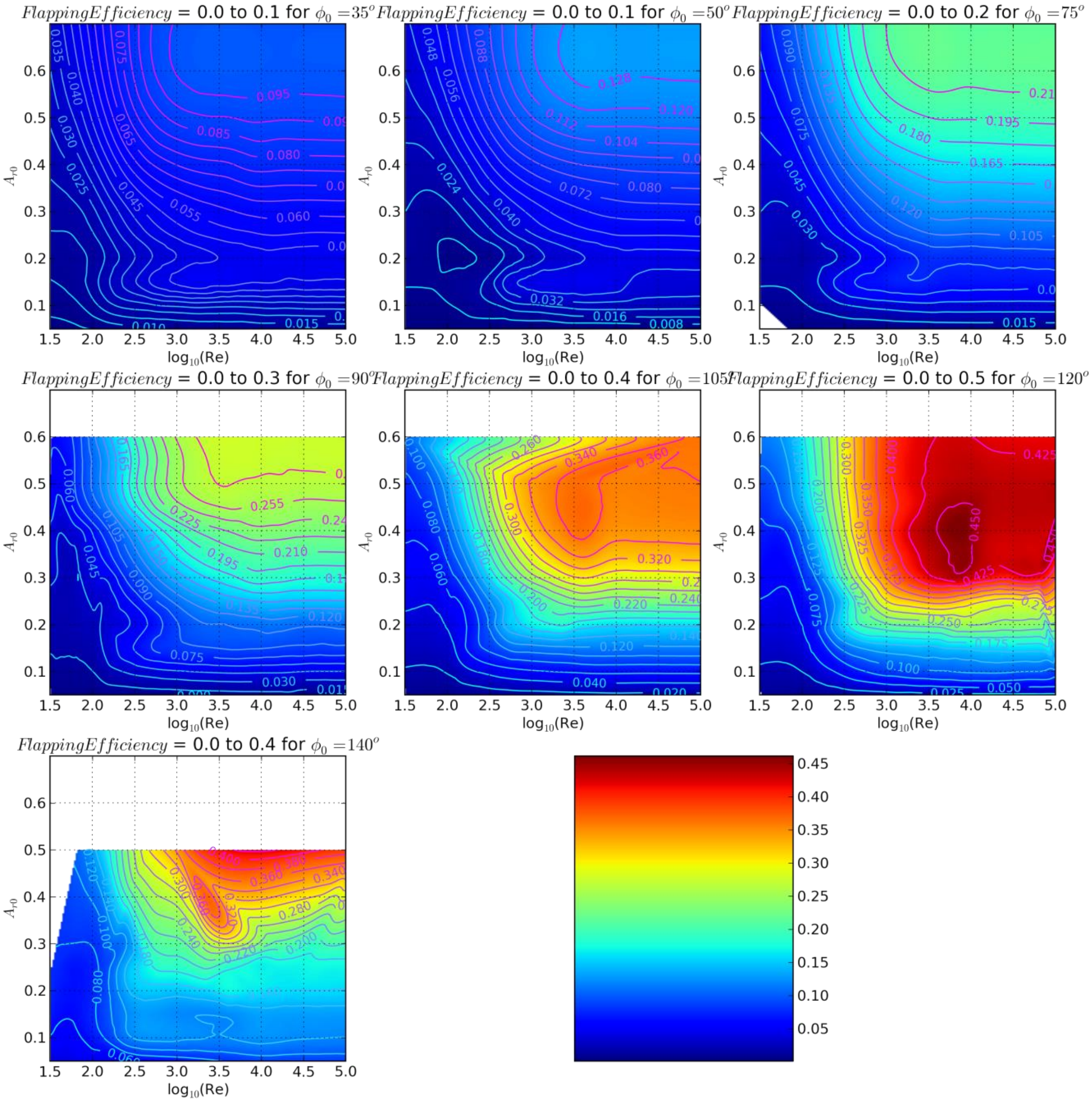


Figure 3-5: Flapping efficiency as a function of Reynolds number and flapping shape

Generally, C_{DP} decreased with Re, leveling off at Re larger than about one hundred (10^2), while C_{DN} increased with increasing Re, leveling off at Re larger than about one thousand (10^3). Similar trends are observed with respect to amplitude, with C_{DN} and C_{DP} generally increasing and decreasing respectively, and leveling off above A_{r0} = 0.4.

Using a symmetry plane at the base of the wing, the initial bending angles of $\phi_0 = 75^o$ and below were simulated at amplitudes as high as $A_{r0} = 0.7$, but the maximum amplitude for $\phi_0 = 90^o$ and above was 0.6 or lower, and as low as 0.25 for $\phi_0 = 140^o$. This was due to the physical constraint imposed by the symmetry wall at the wing's base, as demonstrated in Figure 3-6 through Figure 3-10.

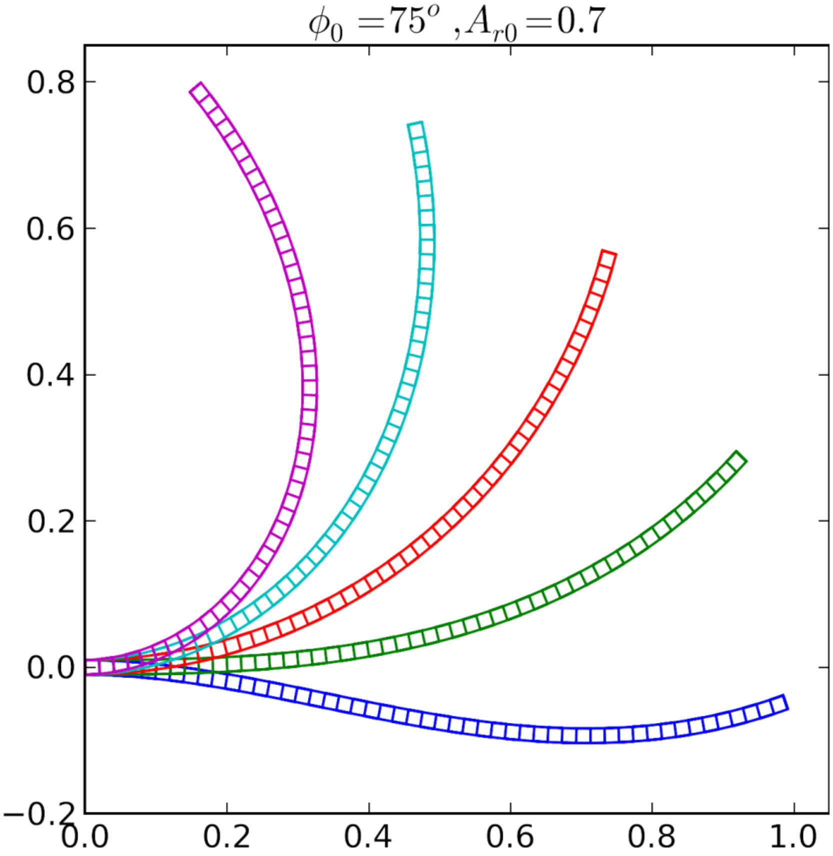


Figure 3-6: Maximum simulated amplitude for initial bending angle $\phi_0 = 75^o$

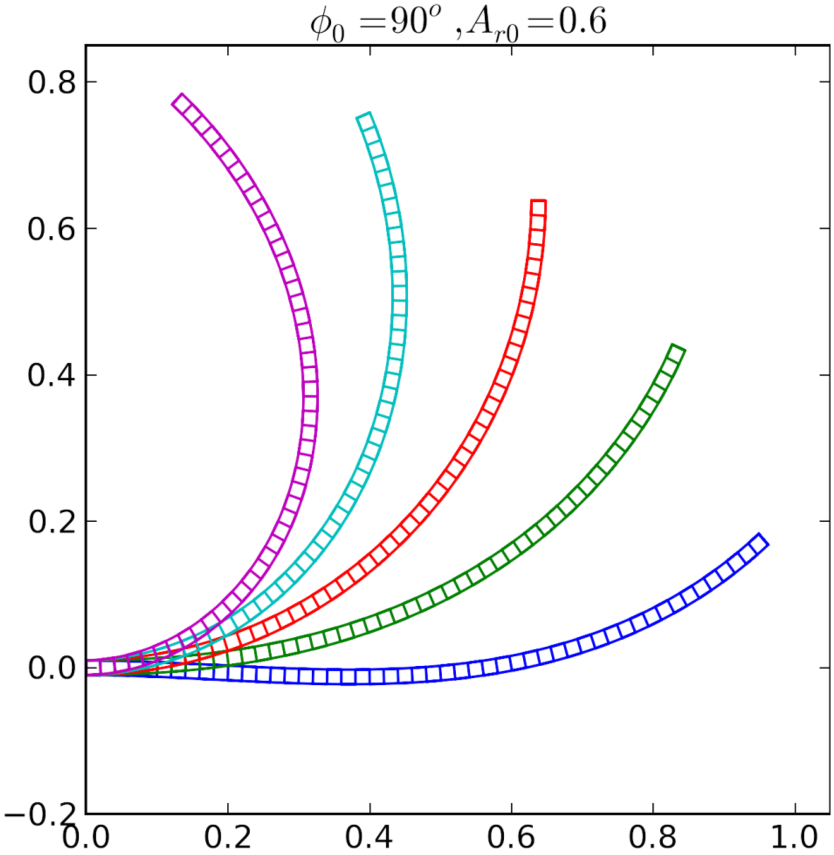


Figure 3-7: Maximum simulated amplitude for initial bending angle ϕ_0 = 90°

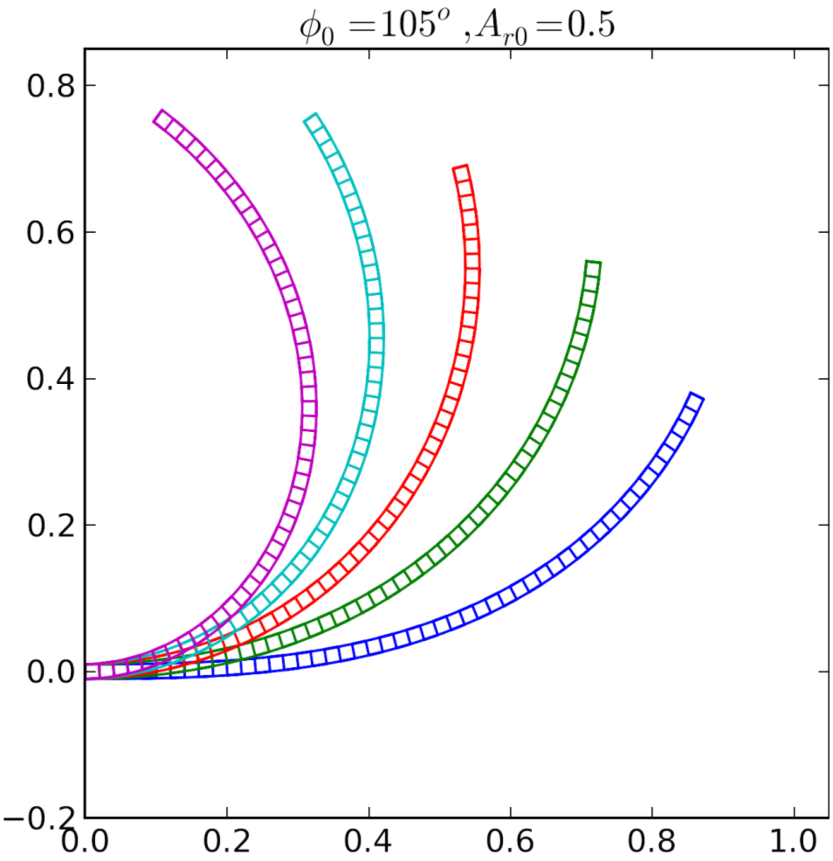


Figure 3-8: Maximum simulated amplitude for initial bending angle ϕ_0 = 105°

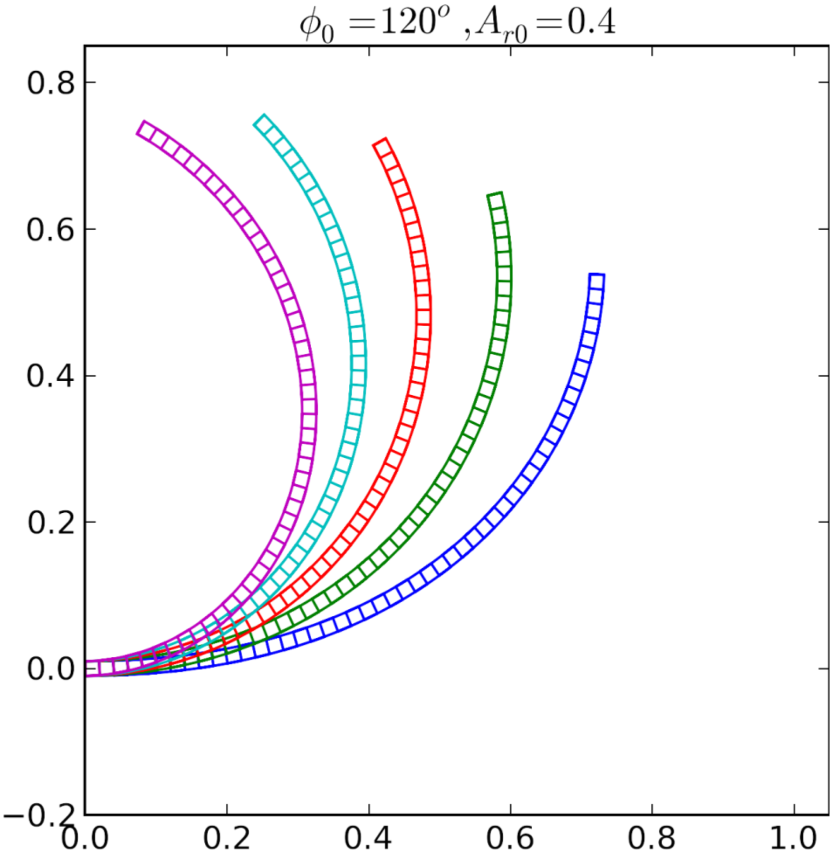


Figure 3-9: Maximum simulated amplitude for initial bending angle ϕ_0 = 120°

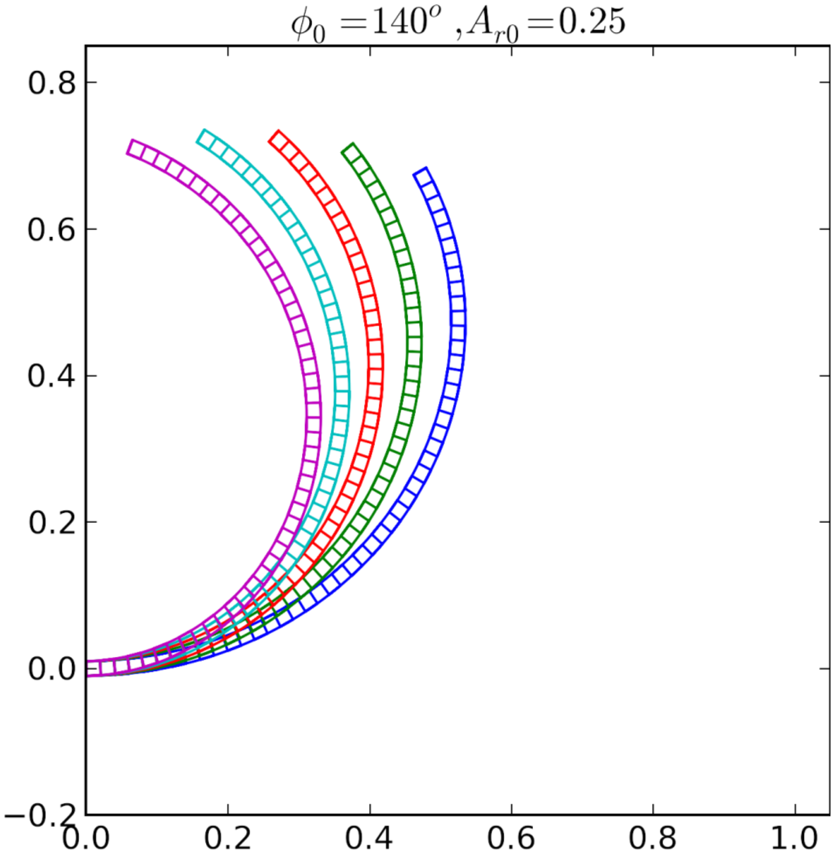


Figure 3-10: Maximum simulated amplitude for initial bending angle $\phi_0 = 140^o$

Some simulations circumvented this limit on amplitude for highly-angled wings by adding a stationary extension to the base of the wings, effectively moving the wings to the right in the figures above and making their tips further from the left walls (note that the other three walls in these pictures are not boundaries; the fluid box continues for 10 wing-lengths in these directions, as explained in Appendix K). This allowed exploring the high-bending angle, high-amplitude regions that would not otherwise be possible. Plotting the maximum relative drag coefficient vs. A_{r0} for a range of Re where these coefficients are steady suggests that the most promising flapping shapes would be otherwise out of reach due to this condition, as shown in Figure 3-11.

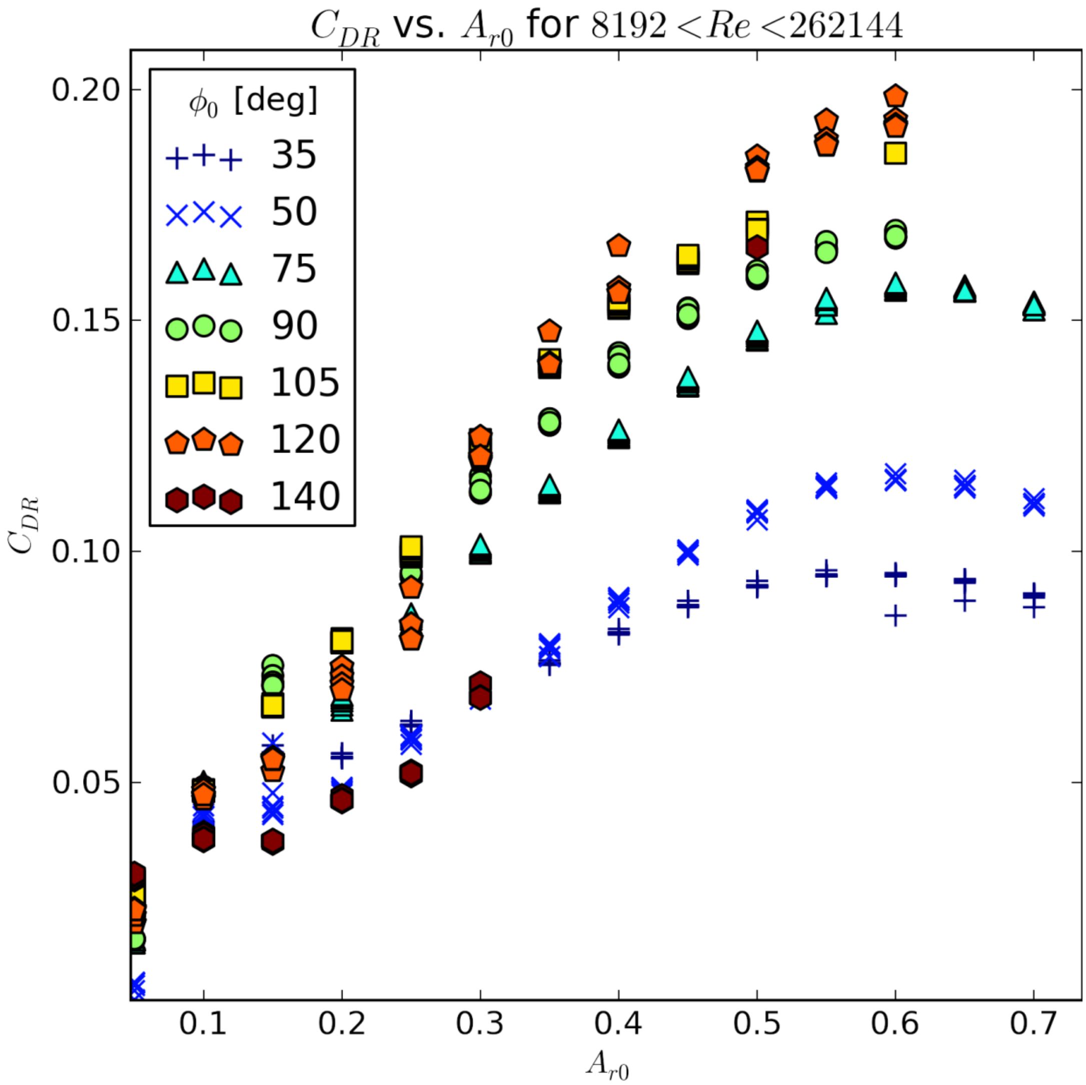


Figure 3-11: Relative drag coefficient vs. amplitude for simulations with Reynolds number between 8192 and 262144.

Notice that for this range of Re, it appears that both ϕ_0 = 105° and 120° will have maximum C_{DR} higher than that of ϕ_0 = 90°. At first glance Figure 3-11 suggests that the optimum C_{DR} at high-Re is for a bending angle somewhere in $105° < \phi_0 < 120°$, but closer inspection shows the curious effect that higher bending angles only have superior C_{DR} at sufficiently high amplitudes, so ϕ_0 = 140° may perform better at larger amplitudes than it appears. Indeed, new data is confirming this idea, as indicated by the data point for $\phi_0 = 140^o$ at $A_{r0} = 0.5$.

Section 3.3.1 Drag Coefficients in Perspective: Dragonfly and Fruit Fly Comparison

Work by Sun et al. in [40] gives us the information we need to determine the equivalent shape and Reynolds number and corresponding drag coefficients for a dragonfly (*Aeschna juncea*) in hovering. According to Sun, dragonflies have a mass of 754 mg, mass specific power of 37 W/kg, wing length of 4.7 cm, flapping frequency of 36 Hz, stroke amplitude (which is actually the peak-to-peak angular distance) of 69^o, and four wings. Approximating each point at position *s* along the wing as moving along an arc with instantaneous position described by $\Phi s \sin(\omega t)$, where Φ is the wing's angular amplitude and ω is its angular frequency, the RMS wing velocity is

$$v_{rms} = \sqrt{\frac{\int_0^L (v_{\max})^2\, ds}{L}} = \sqrt{\frac{\int_0^L (\Phi \omega s)^2\, ds}{L}} = \frac{\Phi \omega L}{\sqrt{3}}$$

From this, we can apply this work's definition of Reynolds number as $\mathrm{Re} \equiv \rho_{air} \frac{v_{rms}\sqrt{A_{wing}}}{\mu}$ to calculate that dragonfly wings operate at Re = 5090. (Note that Sun et al. found Re = 1350 in [40]. This is because their choice of velocity and length scale were the time-average spatial-average velocity and the chord length, both of which have smaller values than my choices, the time-max spatial-rms velocity and the square root of the wing area. It is important to know how Re is defined for a particular application in order to determine force from drag coefficients.)

Knowing that while hovering, the wings must be putting out a force that matches their weight, we can calculate their net drag coefficient, $C_{DN} = 0.504$. Then using their mass specific power and their mass to calculate their power dissipation, we can use their RMS velocity to calculate their power drag coefficient, C_{DP} = 0.514. The ratio of these is their relative drag coefficient, C_{DR} = 0.980.

For comparing with the RS mode, dragonfly wings operate with an effective relative amplitude given by their tip motion per quarter flap cycle divided by its length, which is approximately $A_{r0} = \frac{L\Phi}{L} = \Phi = 0.60$. At this amplitude and Reynolds number, the RS mode generates less net force and requires more power than the dragonfly wings. The specifics depend on which static bending angle wing we compare to, but at best the RS mode generates about half of the force of a dragonfly while simultaneously using twice the power. The best RS mode relative drag coefficient at an amplitude of 0.60 and Reynolds number of 5000 was C_{DR} = 0.198 for ϕ_{0g} = 120°, about 20% as effective as a dragonfly's wings.

Sun et al. also presents information on fruit flies in [40]; fruit flies have wing length L = 0.3 cm, power efficiency P^* = 30W/kg, frequency f = 240 Hz, and stroke amplitude 2Φ = 150°. Combining this with the fruit fly mass of m = 0.3 mg (from [4] and [5]), we can calculate that a fruit fly operates at a Reynolds number of 332 by our definition, and has drag coefficients C_{DN} = 0.094, C_{DP} = 0.084, and C_{DR} of 1.12. While this amplitude is far larger than I was able to simulate, the best relative drag coefficient at a Reynolds number of 330 was about 0.14, only 12% as effective as a fruit fly's wings. Interestingly, this difference is entirely due to power dissipation: the net drag coefficient is superior for the RS mode, about C_{DN} = 0.16, but the power used by the fruit fly was significantly lower than the RS mode's C_{DP} = 1.13. Note that these insects are capable of using their wings to generate more force than their weight, but in doing so they use more than the mass specific power presented here.

However, we need to be careful comparing the drag coefficients that use Reynolds number definitions due to different types of flapping. For the purposes of calculating the

Reynolds number, the insect wings are modelled here as moving like flat plates, with velocity directly proportional to distance from a fixed pivot point. This leads to a different velocity relation than the RS mode, where the velocity is nonlinear in this distance, the region near the tip moving much more than the region near the fixed support.

To circumvent this misleading definition, we can obtain a more direct comparison of the flapping efficiency by finding the power dissipated by RS mode wings of the same length producing the same net force as the insect wings. Doing so gives some easier to appreciate results:

1. Compared to dragonfly wings, L = 4.7 cm resonant cantilever wings that produce 1.9 mN consume 5.5 times as much power; and
2. Compared to fruit fly wings, L = 3 mm resonant cantilever wings that produce 1.5 μN consume 6 times as much power.

Section 3.4 Reversibility at Low Re

At Re $\ll$ 1, fluid motion enters the regime of "Stokes flow", where viscosity effects dominate and the fluid motion is caused only by molecular diffusion (which is random and slow) and momentum diffusion (which is exactly reversible) [41]. Stokes flow is reversible, meaning that wings with symmetric flap cycles generate zero net force: sweeping back along the same path exactly undoes whatever sweeping forward did. This is the flow regime of microorganisms possessing flagella, which get around reversibility by continuously rotating in the same angular direction rather than sweeping back and forth [42].

Referring to Figure 3-12, the simulations indicate that most of the transition to the low-force region occurs as Reynolds number decreases from 1000 to 100, over which the flapping efficiency drops by 80%. Below this, the net force gradually decreases towards a negligible fraction of the average force, which makes calculating the net force from the simulations increasingly difficult and sensitive to errors.

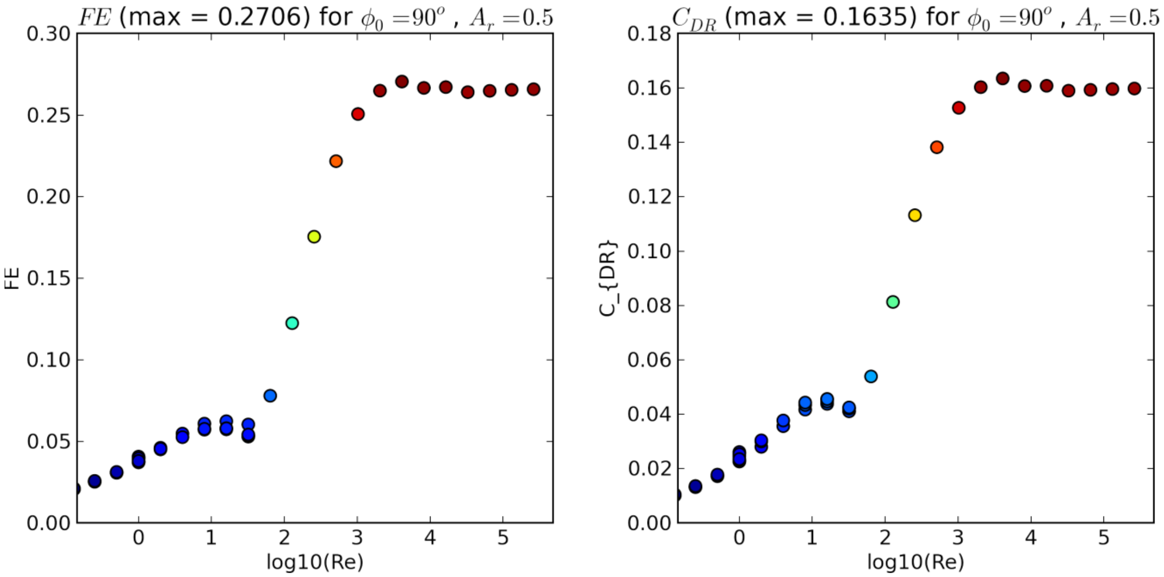


Figure 3-12: Flapping efficiency and relative drag coefficients behaviour in the low Reynolds number regime

Section 3.5 Steady State Drag Coefficients

The plots of net and power drag coefficients from Section 3.3 suggest that both C_{DN} and C_{DP} approach ϕ_0-dependent steady-state values above thresholds in amplitude and Reynolds number. Constant drag coefficients significantly simplify fluid mechanics analysis, so this is worth investigating.

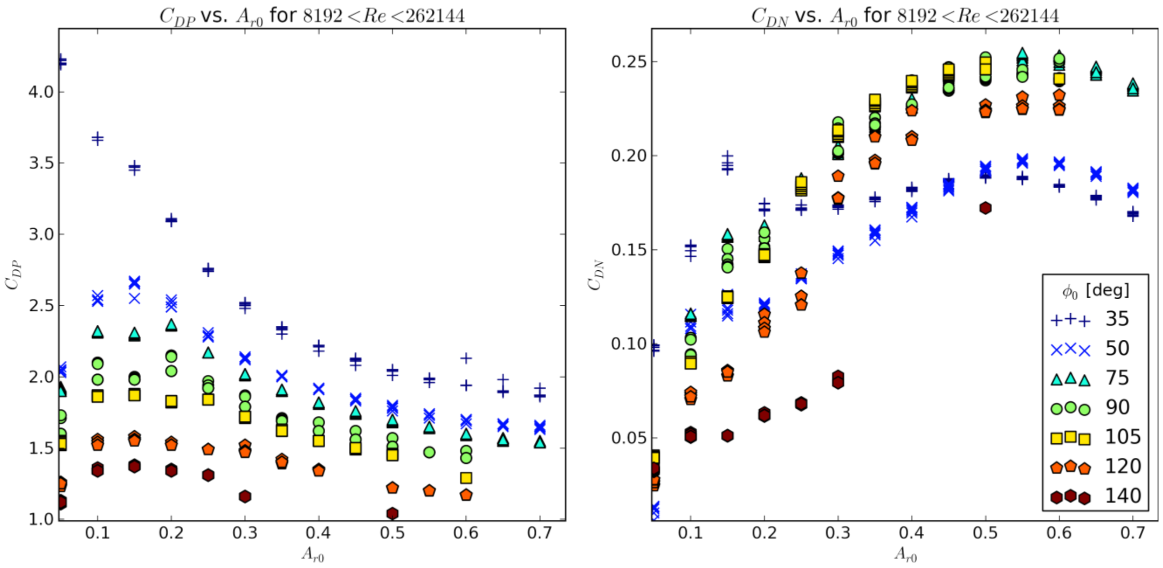


Figure 3-13: Net and power drag coefficient vs. amplitude for Re ≥ 8000

Figure 3-13 plots the drag coefficients vs. amplitude for Re ≥ 8000. In this range of Reynolds number, the net drag coefficient seems to plateau for wing amplitudes between 0.45 and 0.65, while power drag coefficient appears to slowly decrease towards a limiting value as amplitude increases. Even still, the power drag coefficient variability in this range of amplitudes is less than 8% different from its mean value, as shown in Table 3-1.

Table 3-1: Average drag coefficients for Re ≥ 8000, 0.45 ≤ A_r ≤ 0.65

phi0 [deg]	CDN	Stdev	Min	Max	CDP	Stdev	Min	Max
35	0.187	2.10%	-6.91%	2.23%	2.02	3.50%	-6.42%	5.47%
50	0.193	2.60%	-5.23%	3.33%	1.75	3.50%	-5.86%	5.55%
75	0.248	1.90%	-3.99%	3.02%	1.64	4.10%	-5.65%	7.13%
90	0.244	2.20%	-3.52%	3.86%	1.51	3.70%	-5.02%	7.60%
105	0.247	0.88%	-1.29%	1.42%	1.48	1.40%	-2.03%	1.35%

Figure 3-14 plots the drag coefficients vs. Reynolds number for this amplitude range, $0.45 \leq A_r \leq 0.65$. This plot shows that the power and net drag coefficients appear to stabilize for Reynolds numbers above 500 and 2000, respectively. Power drag coefficient appears to have a minimum around Re = $10^{2.5}$, while net drag coefficient has a maximum around $10^{3.7}$, but only for ϕ_0 between 90° and 120°. Unlike the net drag coefficient vs. amplitude, both extrema are close to the limiting value as Re extends to infinity.

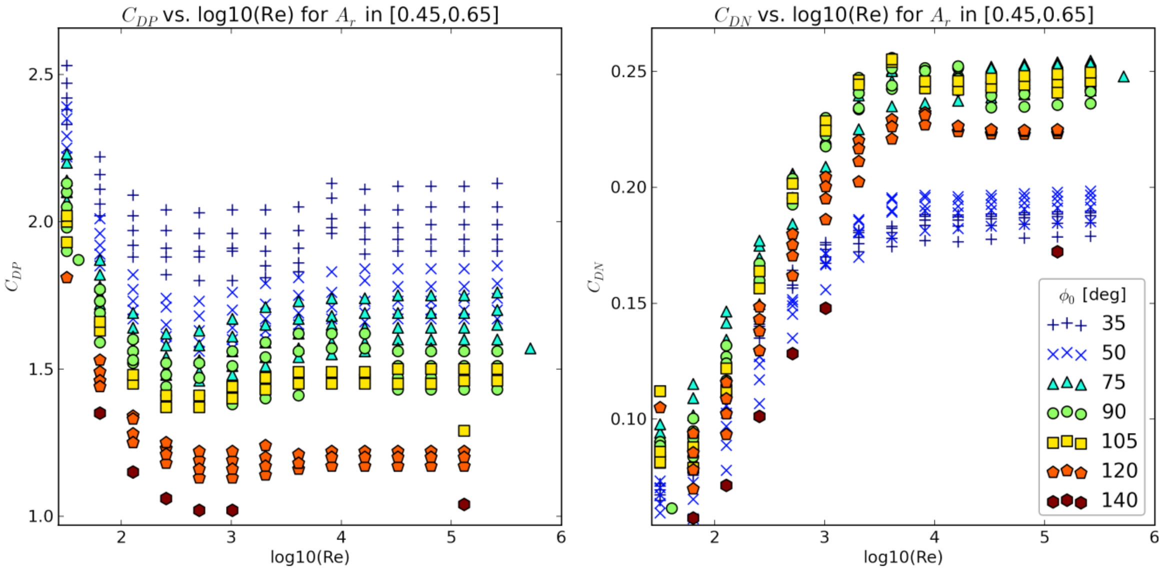


Figure 3-14: Net and power drag coefficients vs. Reynolds number for $0.45 \leq A_r \leq 0.65$

Considering all of these plots, the optimum bending angle for this type of flapping appears to lie in the range of 75° to 105° in terms of net force, with higher bending angles winning in terms of low power dissipation to get this net force.

Chapter 4 The Flying Robot Analysis Framework

The fluid mechanics simulations from Chapter 3 return the net force and power used for each flapping wing given the simulation inputs (ϕ_0, A_{r0}, and Re), and after some math, these are converted to the net drag coefficient and power drag coefficient. Other things equal, the better design is the one with more net force and lower power used. But what if a design change increases the net force *and* power used? Furthermore, even within a fixed set of simulation inputs, which length-frequency combination is best for a given wing-material choice? Most of all, which robot designs would actually fly, which design is best for a given material system, and which material system leads to the best robots?

Evidently the simulation outputs in isolation are not going to answer these questions. For that, we need to consider the simulation results not in isolation but as a piece of a complete "big-picture" robot framework that considers the actual flying robot candidate each simulation would lead to for a given choice of materials. This chapter presents such a framework and some results of applying it to the simulations as a guide in making robot design choices. To clarify, the *analytical framework* is a collection of equations which determine the robot parameters from input stack information, shape information, Reynolds number, materials properties, and drag coefficients. In contrast, the *optimization routine* is a method to use the results of the analytical framework to determine which set of inputs we should use in the first place to produce the most desirable robot.

Section 4.1 is a brief introduction to the framework and associated optimization routine. Section 4.2 presents additional theory and equations necessary to apply the analytical framework algorithm, and section 4.3 combines them to present the complete robot analytical framework. Sections 4.4 and 4.5 implement phase 1 of the optimization routine, applying the analytical framework for all CFX inputs and a particular stack structure, exploring the effect of each constraint. Section 4.6 closes out the chapter by making simplifying assumptions to analytically examine the consequences of using different stack structures and investigate which stack structure parameters are the most important.

Section 4.1 Analytical Framework vs. Optimization Routine

Section 4.1.1 The Analytical Framework

The analytical framework is shown in Figure 4-1. The two input boxes to the analytical framework are the CFX inputs and the stack structure. The CFX inputs include the **flapping shape** (ϕ_0 and A_{r0}) and the Reynolds number, while the **stack structure** means the material properties of the layers making up the wings, including their order, built in strain, and relative thickness to each other (but not the total stack thickness h, which is determined by the analytical framework). From these inputs alone, and a given set of constraints, the analytical framework determines what kind of robot the inputs would lead to, and whether it would be able to fly. For the robots that do fly, the analytical framework calculates the robot’s figures of merit (such as extra power available, how fast the robot moves, etc.).

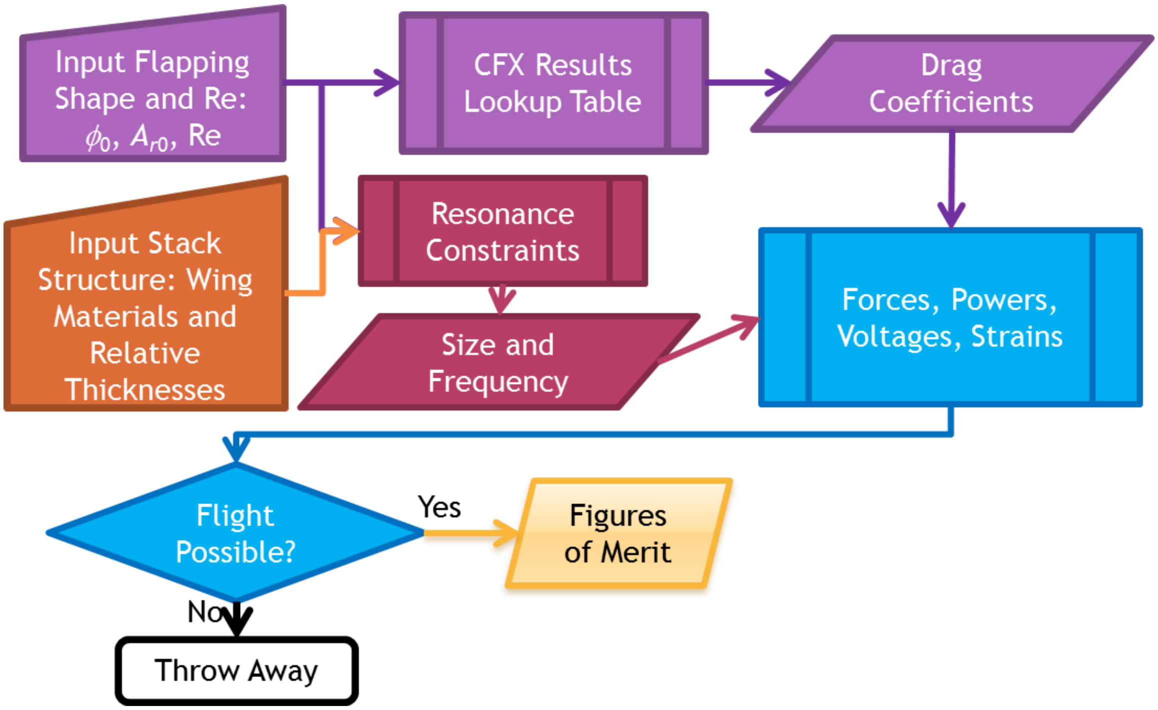

Input Flapping Shape and Re: ϕ_0, A_{r0}, Re
CFX Results Lookup Table
Drag Coefficients
Input Stack Structure: Wing Materials and Relative Thicknesses
Resonance Constraints
Size and Frequency
Forces, Powers, Voltages, Strains
Flight Possible?
Yes
Figures of Merit
No
Throw Away

Figure 4-1: The analytical framework uses constraints to determine which inputs lead to flying robots. For each design that does, the analytical framework also determines that that design's figures of merit.

In the previous chapter, Figure 3-2 and Figure 3-3 presented the net and power drag coefficients of the RS bending mode wings as functions of the CFX inputs (shape and Reynolds number). These plots determine an RS wing's fluidic behaviour over a finely spaced grid of inputs, having mapped the input parameter space in steps of no more than 0.5 in amplitude, 15^o in static bending angle, and factors of 2 in Reynolds number. While Chapter 3's results are interesting in isolation from a fluid mechanics perspective, the analytical framework further capitalizes on the results of this mapping. This chapter will show that when combined with a choice of stack structure, every set of CFX inputs uniquely determines the length, thickness, frequency, and weight of the wings. If further combined with the drag coefficients, these inputs also determine the wings' quality factor, and voltage, net force output, and power dissipation. By using a series of additional constraints, this information also determines whether the resulting robot can fly, and if so, how well it performs. At time of writing, each CFX simulation takes 8 hours on SHARCNET, which would make performing the simulations a serious bottleneck to analyzing robot results. But having mapped the parameter space of the CFX simulations in advance, the analytical framework avoids doing a simulation for each new robot design in favour of looking up previously-determined drag coefficients in a table.

The details of each piece of the analytical framework will be presented in subsequent sections of this chapter.

Section 4.1.2   The Optimization Routine

The optimization routine is twofold. Phase 1 of the optimization routine optimizes the flapping shape and Reynolds number for a given input stack material, as illustrated in Figure 4-2. Phase 2 of the optimization routine optimizes the stack structure by repeating Phase 1 for a number of different possible stack structures, as illustrated in Figure 4-3. This finds the overall optimum robot design.

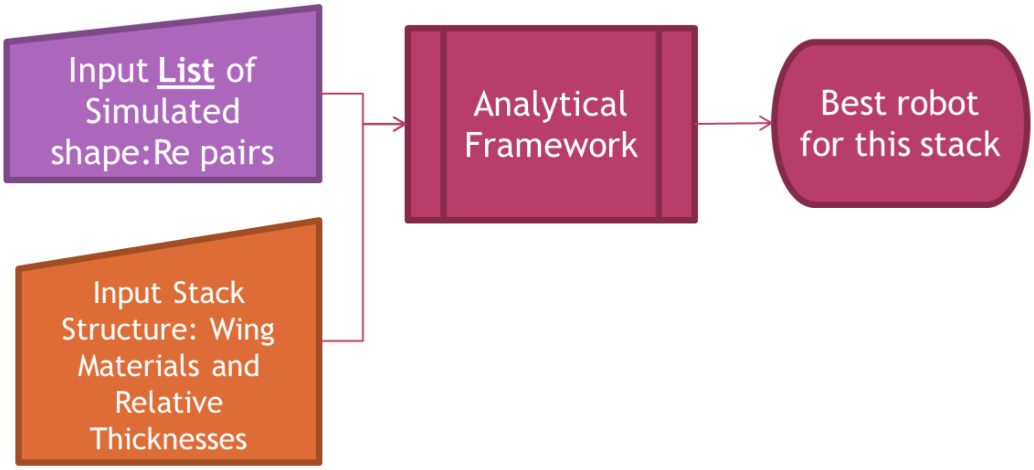

Input List of Simulated shape:Re pairs
Input Stack Structure: Wing Materials and Relative Thicknesses
Analytical Framework
Best robot for this stack

Figure 4-2: Phase 1 of the optimization routine: use list of simulated CFX input parameters to find the best possible robot for a fixed stack structure. This optimization routine is automated.

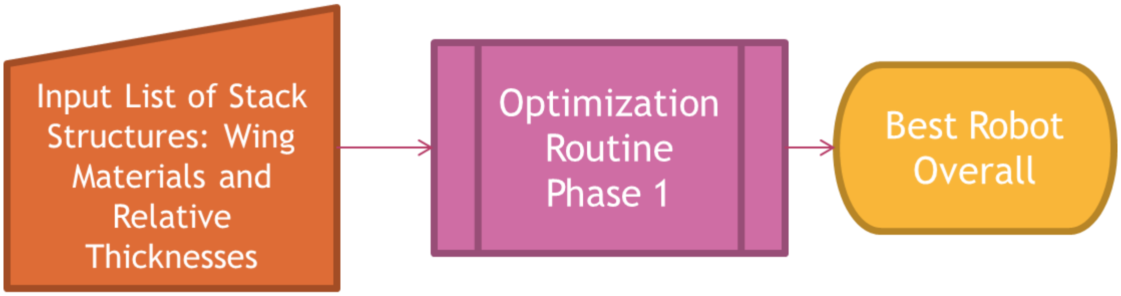

Input List of Stack Structures: Wing Materials and Relative Thicknesses
Optimization Routine Phase 1
Best Robot Overall

Figure 4-3: Phase 2 of the Optimization Routine repeats Phase 1 of the Optimization Routine for a number of different input stack structures to determine the overall best robot.

Currently, Phase 2 is not automated. Future work will involve building a database of possible stack structures to loop over and quickly determine the overall best design for a given choice of optimum (which is dependent on the desired robot application). This necessitates including the individual constraints and challenges of each material.

Section 4.2 Derivations and Details of the Analytical Framework Equations

This section presents the details of the equations necessary to carry out the analytical framework. These equations are collected in

Section 4.2.1 Fixed Stack Structure: Flexural Rigidity and Curvature Revisited

A "fixed stack structure" means:

1. a fixed set of materials,
2. with fixed relative thicknesses to each other, and
3. in a specific order.

For example, Table 4-1 shows the stack structure for a quartz:SU-8 composite wing whose cross-section is shown in Figure 4-4. Layers are numbered from bottom to top, thickness is specified in arbitrary units, and layers with a nonzero "contact number" are electrical contact layers.

Table 4-1: Stack structure example

Layer Number	Material	Contact Number	Thickness [μm]	Built-in Strain
6	SU-8	0	7.600E-06	-0.0075
5	Au	2	1.500E-07	-3.2E-05
4	Cr	2	2.500E-08	-0.00717
3	Quartz	0	7.600E-06	-0.00164
2	Cr	1	2.500E-08	-0.00717
1	Au	1	1.500E-07	-3.2E-05

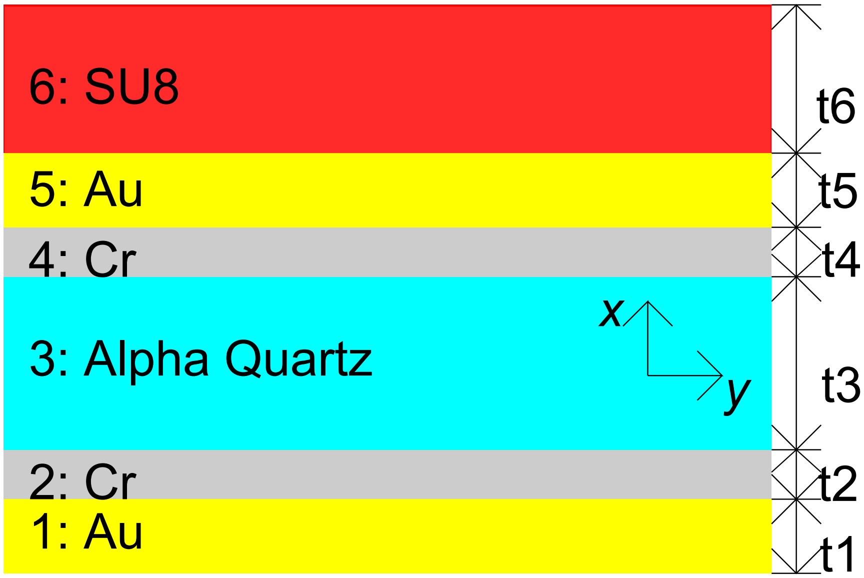


Figure 4-4: Visualized stack structure

By dividing the thickness of each layer t_i by the total thickness *h*, we can compute the relative thickness of each layer $\tilde{t}_i \equiv \frac{t_i}{h}$ as in Table 4-2. Since only relative thicknesses matter for the analytical framework, any stack structure with the same relative thickness of each layer is considered identical to this one.

Table 4-2: Stack structure example with relative thickness numbers

Layer Number	Material	Contact Number	Thickness t_i , [μm]	Relative Thickness, $\tilde{t}_i$	Built-in Strain
6	SU-8	0	7.600E-06	48.875%	-0.0075
5	Au	2	1.500E-07	0.965%	-3.2E-05
4	Cr	2	2.500E-08	0.161%	-0.00717
3	Quartz	0	7.600E-06	48.875%	-0.00164
2	Cr	1	2.500E-08	0.161%	-0.00717
1	Au	1	1.500E-07	0.965%	-3.2E-05

Suppose we further fix the width of the wing to $B = L/4$. Then we can substitute into the multimorph flexural rigidity and curvature formulas from Chapter 2 to obtain simplified formulas that separate the size dependence from the stack structure dependence. Doing so (in the next sections) leads to 4 parameters that depend uniquely on the stack structure and not on the scale of the wings themselves:

1. Effective Young's modulus, E_Y
   a. Definition: $E_Y \equiv \sum \left(E_i \tilde{t}_i^3 + 12 \tilde{t}_i E_i \left(\tilde{y}_i - \frac{\sum \tilde{t}_i E_i \tilde{y}_i}{\sum \tilde{t}_i E_i} \right)^2 \right)$
   b. Interpretation: measures the stack's stiffness. Gives the flexural rigidity via $EI = E_Y \frac{Bh^3}{12}$
2. Average density, ρ_m
   a. Definition: $\rho_m \equiv \sum \rho_{m\_i} \tilde{t}_i$
   b. Interpretation: average measure of the stack's mass per unit volume
3. Curvature constant, γ_{c0}
   a. Definition: $\gamma_{c0} \equiv \frac{\tilde{\mathbf{D}}\tilde{\mathbf{A}}^{-1}}{1 - \frac{1}{2}\tilde{\mathbf{D}}\tilde{\mathbf{A}}^{-1}\tilde{\mathbf{B}}} \tilde{\mathbf{C}}_0$

b. Interpretation: the stack's built-in drive for curvature. Determines the curvature after dividing by thickness: $c_0 = \gamma_{c0} / h$ and the initial bending angle after then multiplying by length: $\phi_0 = \gamma_{c0} \frac{L}{h}$

4. Piezoelectric curvature constant, γ_{cpiezo}

a. Definition: $\gamma_{cpiezo} \equiv \frac{\tilde{\mathbf{D}}\tilde{\mathbf{A}}^{-1}}{1 - \frac{1}{2}\tilde{\mathbf{D}}\tilde{\mathbf{A}}^{-1}\tilde{\mathbf{B}}} \tilde{\mathbf{C}}_{piezo}$

b. Interpretation: Measures the ease with which voltage transduces into wing bending; this is a curvature constant per unit electric field,

$$c_{piezo} = \frac{\gamma_{cpiezo}}{h} \frac{V}{h} = \gamma_{cpiezo} \frac{V}{h^2}$$

Section 4.2.1.1 Flexural Rigidity

Recall that flexural rigidity of a multimorph was, from equation (2.1)

$$EI = \sum \left(E_i I_{0i} + A_i E_i \left(y_i - \bar{Y} \right)^2 \right)$$

where

1. $\bar{Y}$ is the location of the *AE* centroid: $\bar{Y} \equiv \frac{\sum A_i E_i y_i}{\sum A_i E_i}$,

2. y_i is the vertical centre of mass of the i^{th} layer, and

3. I_{0i} is the moment of inertia of the i^{th} rectangular section about its centroid: $I_{0i} = \frac{Bt_i^3}{12}$

Re-expressing in terms of relative layer positions and thicknesses:

1. $A_i = Bt_i = \frac{L}{4}\tilde{t}_i h$

2. $y_i = \tilde{y}_i h$ (where $\tilde{y}_i \equiv \frac{y_i}{h}$ is the dimensionless layer centroid position)

a. $\therefore \bar{Y} = \frac{\sum \tilde{t}_i E_i \tilde{y}_i}{\sum \tilde{t}_i E_i} h$

3. $I_{0i} = \frac{Bt_i^3}{12} = \frac{\tilde{t}_i^3}{48} Lh^3$

$$\therefore EI = \sum \left(E_i I_{0i} + A_i E_i \left(y_i - \bar{Y} \right)^2 \right)$$

$$= \sum \left(E_i \frac{\tilde{t}_i^3}{48} Lh^3 + \frac{L}{4} \tilde{t}_i h E_i \left(\tilde{y}_i h - \frac{\sum \tilde{t}_i E_i \tilde{y}_i}{\sum \tilde{t}_i E_i} h \right)^2 \right)$$

$$= \left[\sum \left(E_i \frac{\tilde{t}_i^3}{48} + \frac{\tilde{t}_i}{4} E_i \left(\tilde{y}_i - \frac{\sum \tilde{t}_i E_i \tilde{y}_i}{\sum \tilde{t}_i E_i} \right)^2 \right) \right] Lh^3$$

Compare this with the flexural rigidity of a single-material beam of width $B = L/4$ and thickness h: $EI = E\frac{Bh^3}{12} = E\frac{Lh^3}{48}$, where E is the Young's modulus. Therefore, define the stack's "effective Young's modulus" E_Y such that:

$$E_Y \frac{Lh^3}{48} = \left[\sum \left(E_i \frac{\tilde{t}_i^3}{48} + \frac{\tilde{t}_i}{4} E_i \left(\tilde{y}_i - \frac{\sum \tilde{t}_i E_i \tilde{y}_i}{\sum \tilde{t}_i E_i} \right)^2 \right) \right] Lh^3$$

$$E_Y \equiv \sum \left(E_i \tilde{t}_i^3 + 12 \tilde{t}_i E_i \left(\tilde{y}_i - \frac{\sum \tilde{t}_i E_i \tilde{y}_i}{\sum \tilde{t}_i E_i} \right)^2 \right)$$

With this definition, the flexural rigidity of the entire stack is $EI = E_Y \frac{Bh^3}{12} = E_Y \frac{Lh^3}{48}$.

Section 4.2.1.2 Resonant Frequency

$$f_{res} = \frac{\lambda^2}{L^2 2\pi} \sqrt{\frac{EI}{Bh\rho_m}} f_{resCor}$$

$$= \frac{\lambda^2}{L^2 2\pi} \sqrt{\frac{E_Y \frac{Bh^3}{12}}{Bh\rho_m}} f_{resCor}$$

$$= \frac{\lambda^2}{2\pi} \sqrt{\frac{E_Y}{12\rho_m}} f_{resCor} \frac{h}{L^2}$$

$\frac{E_Y}{\rho_m}$ depends only on the stack structure (the density is a thickness-weighted average), but

f_{resCor} depends only on the initial bending angle (i.e., the CFX simulation chosen).

Section 4.2.1.3 Curvature Parameters

$$\mathbf{D} = \frac{1}{\sum_{i=1}^{m} E_i I_i} \begin{bmatrix} \frac{t_1}{2} & t_1 + \frac{t_2}{2} & \cdots & \sum_{j=1}^{m-1} t_j + \frac{t_m}{2} \end{bmatrix}$$

Using the definition of y_i as the location of the i^{th} layer's centroid from the flexural rigidity section, $\mathbf{D} = \frac{1}{\sum_{i=1}^{m} E_i I_i} \begin{bmatrix} y_1 & y_2 & \cdots & y_m \end{bmatrix}$. Further substituting the formula for second moment of area of the i^{th} layer, $I_i = \frac{B_i t_i^3}{12}$,

$$\mathbf{D} = \frac{1}{\sum_{i=1}^{m} E_i \frac{B_i t_i^3}{12}} \begin{bmatrix} y_1 & y_2 & \cdots & y_m \end{bmatrix}$$, and now writing in terms of the relative parameters:

$$\begin{aligned} \mathbf{D} &= \frac{4}{Lh^3 \sum E_i \frac{\tilde{t}_i^3}{12}} \begin{bmatrix} \tilde{y}_1 & \tilde{y}_2 & \cdots & \tilde{y}_m \end{bmatrix} h \\ &= \frac{1}{\sum E_i \frac{\tilde{t}_i^3}{12}} \begin{bmatrix} \tilde{y}_1 & \tilde{y}_2 & \cdots & \tilde{y}_m \end{bmatrix} \frac{4}{Lh^2} \\ &\equiv \tilde{\mathbf{D}} \frac{4}{Lh^2} \end{aligned}$$

$$\mathbf{A}=\begin{bmatrix} \frac{1}{A_1E_1} & \frac{-1}{A_2E_2} & 0 & \dots \\ 0 & \frac{1}{A_2E_2} & -\frac{1}{A_3E_3} & \dots \\ 0 & 0 & \dots & \dots \\ \dots & 0 & \frac{1}{A_{m-1}E_{m-1}} & \frac{-1}{A_mE_m} \\ 1 & 1 & 1 & 1 \end{bmatrix}=\frac{4}{Lh}\begin{bmatrix} \frac{1}{\tilde{t}_1E_1} & \frac{-1}{\tilde{t}_2E_2} & 0 & \dots \\ 0 & \frac{1}{\tilde{t}_2E_2} & -\frac{1}{\tilde{t}_3E_3} & \dots \\ 0 & 0 & \dots & \dots \\ \dots & 0 & \frac{1}{\tilde{t}_{m-1}E_{m-1}} & \frac{-1}{\tilde{t}_mE_m} \\ 1 & 1 & 1 & 1 \end{bmatrix}\equiv\frac{4}{Lh}\tilde{\mathbf{A}}$$

$$\mathbf{B}=\begin{bmatrix} t_1+t_2 \\ t_2+t_3 \\ \dots \\ t_{m-1}+t_m \\ 0 \end{bmatrix}=\begin{bmatrix} \tilde{t}_1+\tilde{t}_2 \\ t_2+t_3 \\ \dots \\ \tilde{t}_{m-1}+\tilde{t}_m \\ 0 \end{bmatrix}h=\tilde{\mathbf{B}}h$$

and $\mathbf{C}_0=\begin{bmatrix} \varepsilon_{free0\_2}-\varepsilon_{free0\_1} \\ \varepsilon_{free0\_3}-\varepsilon_{free0\_2} \\ \dots \\ \varepsilon_{free0\_m}-\varepsilon_{free0\_m-1} \\ 0 \end{bmatrix}$ is already relative to the stack structure only.

The curvature is then:

$$c \equiv \frac{1}{\rho} = \frac{\mathbf{DA}^{-1}}{1-\frac{1}{2}\mathbf{DA}^{-1}\mathbf{B}}\mathbf{C}$$

$$= \frac{\left(\frac{4}{Lh^2}\tilde{\mathbf{D}}\right)\left(\frac{4}{Lh}\tilde{\mathbf{A}}\right)^{-1}}{1-\frac{1}{2}\left(\frac{4}{Lh^2}\tilde{\mathbf{D}}\right)\left(\frac{4}{Lh}\tilde{\mathbf{A}}\right)^{-1}\left(h\tilde{\mathbf{B}}\right)}\mathbf{C}$$

$$= \frac{\left(\frac{4}{Lh^2}\tilde{\mathbf{D}}\right)\left(\frac{Lh}{4}\tilde{\mathbf{A}}^{-1}\right)}{1-\frac{1}{2}\left(\frac{4}{Lh^2}\tilde{\mathbf{D}}\right)\left(\frac{Lh}{4}\tilde{\mathbf{A}}^{-1}\right)\left(h\tilde{\mathbf{B}}\right)}\mathbf{C}$$

$$= \frac{\tilde{\mathbf{D}}\tilde{\mathbf{A}}^{-1}\frac{1}{h}}{1-\frac{1}{2}\tilde{\mathbf{D}}\tilde{\mathbf{A}}^{-1}\tilde{\mathbf{B}}}\mathbf{C}$$

where $\tilde{\mathbf{D}} \equiv \frac{1}{\sum E_i \frac{\tilde{t}_i^3}{12}}\left[\tilde{y}_1 \quad \tilde{y}_2 \quad \cdots \quad \tilde{y}_m\right]$, $\tilde{\mathbf{A}} \equiv \begin{bmatrix} \frac{1}{\tilde{t}_1 E_1} & \frac{-1}{\tilde{t}_2 E_2} & 0 & \cdots \\ 0 & \frac{1}{\tilde{t}_2 E_2} & -\frac{1}{\tilde{t}_3 E_3} & \ddots \\ 0 & 0 & \ddots & \\ \cdots & 0 & \frac{1}{\tilde{t}_{m-1} E_{m-1}} & \frac{-1}{\tilde{t}_m E_m} \\ 1 & 1 & 1 & 1 \end{bmatrix}$, and

$$\tilde{\mathbf{B}} \equiv \begin{bmatrix} \tilde{t}_1 + \tilde{t}_2 \\ \tilde{t}_2 + \tilde{t}_3 \\ \vdots \\ \tilde{t}_{m-1} + \tilde{t}_m \\ 0 \end{bmatrix}.$$

Built-in Curvature:

The built-in curvature is now $c_0 = \frac{\mathbf{DA}^{-1}}{1-\frac{1}{2}\mathbf{DA}^{-1}\mathbf{B}}\mathbf{C}_0 = \frac{\tilde{\mathbf{D}}\tilde{\mathbf{A}}^{-1}}{1-\frac{1}{2}\tilde{\mathbf{D}}\tilde{\mathbf{A}}^{-1}\tilde{\mathbf{B}}}\frac{1}{h}\tilde{\mathbf{C}}_0 \equiv \gamma_{c0}\frac{1}{h}$, where the curvature constant $\gamma_{c0} \equiv \frac{\tilde{\mathbf{D}}\tilde{\mathbf{A}}^{-1}}{1-\frac{1}{2}\tilde{\mathbf{D}}\tilde{\mathbf{A}}^{-1}\tilde{\mathbf{B}}}\tilde{\mathbf{C}}_0$

Interpretation: A curvature of c = 1 rad/m means that in 1 m the beam subtends an angle of 1 rad. Since $\gamma_{c0} = c_0{*}h$, this parameter is not itself a curvature but rather a curvature times a wing thickness. So, a γ_{c0} of 1μm-rad/mm means that a 1 μm-thick beam subtends a 1 rad angle over 1 mm. Where thickness is a wing's resistance to curvature, γ_{c0} is its drive.

Curvature from the piezoelectric effect:

Suppose the stack is a unimorph with layer #2 as a piezoelectric (layer #1 is a contact layer). Then:

$$\mathbf{C}_{piezo} = \begin{bmatrix} \varepsilon_{piezo2} \\ -\varepsilon_{piezo2} \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

where the piezoelectric strain is proportional to the appropriate piezoelectric coupling coefficient (say, d_{31}) and the field: $\varepsilon_{piezo2} = d_{31,2}E_{3,2}$, where crystal direction 3 is the thickness direction, the indices before the comma refer to piezoelectric crystal directions, and those after the comma refer to the stack layer number. The field is given by the voltage divided by the thickness: $E_{3,2} = \frac{V}{t_2}$, and so the piezo strain matrix can be written:

$$\mathbf{C}_{piezo}=\begin{bmatrix}\varepsilon_{piezo2}\\-\varepsilon_{piezo2}\\0\\\vdots\\0\end{bmatrix}=\begin{bmatrix}d_{31,2}\dfrac{V}{t_2}\\-d_{31,2}\dfrac{V}{t_2}\\0\\\vdots\\0\end{bmatrix}=\begin{bmatrix}d_{31,2}/\tilde{t}_2\\-d_{31,2}/\tilde{t}_2\\0\\\vdots\\0\end{bmatrix}\frac{V}{h}\equiv\tilde{\mathbf{C}}_{piezo}\frac{V}{h}$$

and the piezo-induced curvature is then:

$$c_{piezo}=\frac{\mathbf{DA}^{-1}}{1-\frac{1}{2}\mathbf{DA}^{-1}\mathbf{B}}\mathbf{C}_{piezo}=\frac{\tilde{\mathbf{D}}\tilde{\mathbf{A}}^{-1}}{1-\frac{1}{2}\tilde{\mathbf{D}}\tilde{\mathbf{A}}^{-1}\tilde{\mathbf{B}}}\frac{1}{h}\tilde{\mathbf{C}}_{piezo}\frac{V}{h}\equiv\gamma_{cpiezo}\frac{V}{h^2}$$

Section 4.2.1.4 Stack Structure Examples

Following are three examples of piezoelectric stacks (“stack structures”) for use as inputs for the analytical framework. It is necessary to select relative layer thicknesses and the built-in layer strains.

1. Quartz-Silicon unimorph with chrome/gold/tin contact layers
    a. Single crystal Si passive structural layer (solar cell; 0.2% compressive compared to the Quartz, 6.2 μm)
    b. Gold/Tin eutectic bond layer; eutectic @ 280^{o}C (50 nm)
    c. Chromium layer (for adhesion to quartz) (5 nm)
    d. Single-crystal *x*-cut quartz active piezoelectric layer (tensile) (8.1 μm)
    e. Chromium, 5 nm
    f. Back gold contact, 50 nm
2. Quartz-SU8 unimorph with chrome/gold contact layers
    a. SU-8 layer (0.75% tensile) (7.6 μm)
    b. Gold contact layer (150 nm)
    c. Chromium layer (for adhesion to quartz) (25 nm)

   d. Single-crystal x-cut quartz (active piezoelectric layer, compressive) (7.6 μm)
   e. Chromium, 25 nm
   f. Back gold contact (150 nm)
3. InGaP-InGaP bimorph with GaAs contact layers
   a. GaAs contact layer, degenerate n-doped 30 nm
   b. $Ga_xIn_{1-x}P$ piezoelectric layer, 0.1% tensile, undoped, 750 nm
   c. GaAs contact layer, degenerate p-doped 40 nm
   d. $Ga_xIn_{1-x}P$ piezoelectric layer, 0.1% compressive, undoped, 750 nm
   e. GaAs contact layer, degenerate n-doped 30 nm

These stack structures lead to the parameters shown in Table 4-3. This indicates that the stacks have some key differences. The InGaP stack is the stiffest, but also the densest. For any given length and thickness, the Q:SU-8 stack will have the highest bending angle because it has the highest curvature constant, but will also be more pliant due to its low Young's modulus.

Table 4-3: Stack parameters for three multimorph stacks

Stack	γ_{c0}	E_Y	ρ_m	γ_{cpiezo}
	μm-rad /mm	GPa	kg/m^3	μm^2-mrad /mm-V
Q:Si	3.446	110.8	2631	6.397
Q:SU8	3.929	19.04	2273	3.052
InGaP	2.97	136.1	4518	2.294

The difference in γ_{cpiezo} between the Q:Si and Q:SU8 stacks is due to poor stiffness matching between the quartz and SU-8 compared to the quartz and silicon, as explained in Section 2.1. In contrast, the InGaP design is perfectly stiffness matched, and so has a comparable γ_{cpiezo} despite having a far inferior effective piezoelectric coefficient: $d_{piezo}' = 0.385\text{pm/V}$ for InGaP compared to 2.44pm/V for quartz.

Section 4.2.2 Q, Quality Factor

Insects flap their wings at a frequency different from that of their wings, instead flapping at the resonant frequency of their thorax (the muscles of which are the "actuator") with the wings just along for the ride [43]. In contrast, the fluid diode cantilever wings are (quite uniquely) both the wings and the actuator, so working at the resonant frequency of one means working at the resonant frequency of both. There is good reason to do this. At resonance, the amplitude A_{r0} is increased compared to the low-frequency (or "DC") amplitude the same driving force would produce A_{r0DC}:

$$A_{r0} = QA_{r0DC}$$

where Q is a dimensionless quantity called the "quality factor".

It turns out this "amplitude amplification" definition for the quality factor is not the only one. As you can see from a derivation of the basic forced-response of a linear system, presented in Appendix M, the quality factor is also equal to the ratio of stored to dissipated energy per cycle:

$$Q = \frac{2\pi U_{\max}}{T \times P_A}$$

where $U_{\max}$ is the peak stored mechanical energy, T is the actuation period, and P_A is the average power dissipated into damping.

This dual nature of the quality factor provides another constraint equation and a method to calculate the predicted amplitude of the wings in resonance given information about the rate of energy dissipation into the fluid (which we know from the CFX simulations).

The analytical framework assumes that the damping is entirely fluidic. This turns out to be justified if the mechanical quality factor (the one obtained by operating the resonator in vacuum) is much larger than the quality factor in air.

Section 4.2.2.1 DC Response of the Wing From Voltage: Relation between Ar0 and Phi

In DC, the bending shape of the wing in response to voltage is not actually the RS mode at all, but rather a circular arc with an angle of $\phi = \phi_0 + \Phi_{DC}$, so that a slowly varying (approximately DC compared to resonance) voltage leads to an angle amplitude Φ_{DC} resulting from the curvature amplitude C_{DC} through the regular multimorph static bending angle formulas from Section 2.2.2. This means that piezoelectric actuation at the resonant frequency excites the RS mode using a constant-curvature bending shape. As such, the equivalent A_{r0DC} to use in the quality factor definition that $A_{r0} = QA_{r0DC}$ is the one resulting from *projecting* the DC constant-curvature bending shape onto the RS bending mode. This involves knowing the infinite set of orthogonal resonant shapes and writing the RS Bending as a superposition of them all.

Section 4.2.2.2 Background Information on Curvature from the Two Modes

For constant-curvature bending (i.e., the low-frequency bending), a curvature change leads directly to a wing bending angle change: $\Phi \equiv \Delta\phi = L\Delta c$ (since Δc is a constant, there is no need to evaluate it anywhere in particular along the length).

For RS bending in contrast, the curvature change is not a constant but instead is the derivative of the bending angle expression,

$$\Delta\beta \equiv A_r \frac{\lambda}{2}\left[\sinh\frac{\lambda s}{L} + \sin\frac{\lambda s}{L} + \frac{\cos\lambda + \cosh\lambda}{\sin\lambda + \sinh\lambda}\left(\cos\frac{\lambda}{L}s - \cosh\frac{\lambda}{L}s\right)\right]$$

Section 4.2.2.3 Approximation by Tip Displacement

The answer we get for relating A_{r0DC} to Φ_{DC} should be similar to the one we get by equating the tip displacements of each mode. The tip displacement for a small amplitude

RS bending is $w(L)=\int_0^L \Delta\beta ds = A_r L$, while for constant-curvature bending it is

$$w(L)=\int_0^L s\frac{\Phi}{L}ds=\frac{\Phi L}{2}.$$

Therefore, at small amplitudes, constant-curvature bending with an angle amplitude of Φ has the same tip displacement as RS bending with $A_{r0}=\frac{\Phi}{2}$.

If we use this to set the A_{r0} to the one that gives the same tip displacement:

$$\boxed{A_{r0}=QA_{rDC}=Q\frac{\Phi_{DC}}{2}}$$

Section 4.2.2.4 Approximation by Projecting the Flat Cantilever Modes

Since we do not know all the modes of the curved beam, we can appeal to the flat beam and assume that the same fraction of its first mode that is used to represent a constant-curvature excitation also applies to a curved beam.

For an initially flat beam, the built-in curvature is $c_0=0$, and so the extra angle is related to the total curvature: $\Phi=L\Delta c=Lc$. At small amplitude, the curvature of a flat beam is approximately the second derivative with respect to distance from the fixed end: $c\approx w_{xx}$. Therefore, a small constant curvature bending angle of Φ means a displacement from a flat beam of

$$w=\int_0^x\int_0^x cdxdx$$

$$w=\frac{\Phi}{2L}x^2$$

Let the set of all of the resonant mode shapes be $\left\{w_i\left(x\right)\right\}_{i=1..\infty}$. Then we can represent any driving shape as a superposition of them:

$$\frac{\Phi}{2L}x^2 = \sum_{i=1}^{\infty} b_i w_i(x)$$

Since the mode shapes of the flat beam form an orthogonal set, we can find the contribution from the first mode b_1 by multiplying both sides of the equation and integrating:

$$\int_0^L \frac{\Phi}{2L}x^2 w_1(x)\,dx = \int_0^L \sum_{i=1}^{\infty} b_i w_i(x) w_1(x) dx$$

$$\frac{\Phi}{2L}\int_0^L x^2 w_1(x)\,dx = b_1 \int_0^L (w_1(x))^2\,dx$$

$$\therefore b_1 = \frac{\Phi}{2L}\frac{\int_0^L x^2 w_1(x)\,dx}{\int_0^L (w_1(x))^2\,dx}$$

Using the known

$w_1(x) = \int_0^x \frac{\Delta\beta(x)}{A_{r0}}dx = \frac{L}{2}\left[\cosh\frac{\lambda x}{L} - \cos\frac{\lambda x}{L} + \frac{\cos\lambda + \cosh\lambda}{\sin\lambda + \sinh\lambda}\left(\sin\frac{\lambda x}{L} - \sinh\frac{\lambda x}{L}\right)\right]$ (divided out the A_{r0} from the previous $\Delta\beta$ definition because here we just want the mode shape; the amplitude A_{r0} is then equal to b_1), and the nondimensionalizing substitution $u \equiv \frac{x}{L}$ these integrals become:

$$\int_0^L x^2 w_1(x)\,dx = \int_0^1 (L^2u^2)\frac{L}{2}\left(\cosh\lambda u - \cos\lambda u + \frac{\cos\lambda + \cosh\lambda}{\sin\lambda + \sinh\lambda}(\sin\lambda u - \sinh\lambda u)\right)(Ldu)$$

$$= L^4\frac{1}{2}\int_0^1 u^2\left(\cosh\lambda u - \cos\lambda u + \frac{\cos\lambda + \cosh\lambda}{\sin\lambda + \sinh\lambda}(\sin\lambda u - \sinh\lambda u)\right)du$$

$$\equiv L^4\frac{1}{2}(0.44539...)$$

and

$$\int_0^L \left(w_1(x)\right)^2 dx = \int_0^1 \left(\frac{L}{2}\right)^2 \left(\cosh\lambda u - \cos\lambda u + \frac{\cos\lambda + \cosh\lambda}{\sin\lambda + \sinh\lambda}\left(\sin\lambda u - \sinh\lambda u\right)\right)^2 \left(Ldu\right)$$

$$= L^3 \left(\frac{1}{2}\right)^2 \int_0^1 \left(\cosh\lambda u - \cos\lambda u + \frac{\cos\lambda + \cosh\lambda}{\sin\lambda + \sinh\lambda}\left(\sin\lambda u - \sinh\lambda u\right)\right)^2 du$$

$$\equiv L^3 \left(\frac{1}{2}\right)^2 (1)$$

Substituting, we find the first mode amplitude component is

$b_1 = \dfrac{\Phi}{2L} \dfrac{\int_0^L x^2 w_1(x)\,dx}{\int_0^L \left(w_1(x)\right)^2 dx} = \dfrac{\Phi}{2L} \dfrac{\left(L^4 \frac{1}{2} I_1\right)}{\left(L^3 \left(\frac{1}{2}\right)^2 I_2\right)} = \Phi I_1 \approx 0.445$. Therefore, a cylindrical bending amplitude angle of Φ_{DC} should lead to an RS mode amplitude of:

$$\boxed{A_{r0DC} = 0.445\Phi_{DC}}$$

As expected, this is close to the equivalent tip-displacement relation between these bending mode shapes from the previous section, which found $A_{r0DC} = 0.5\Phi_{DC}$.

Section 4.2.3 RS Velocity-Kinetic Energy Relations

Referring to the RS mode potential and kinetic energy derivations in Appendix I, we can determine the quality factor constraint equation for the RS bending mode:

Defined the time-max spatial-rms velocity as $v_{rms} \equiv \sqrt{\dfrac{\int_0^L v_{\max}^2 ds}{L}}$

then found that

$$\boxed{v_{rms}\left(\phi_0\right) = \frac{v_{rms}(0)}{f_{resCor}\left(\phi_0\right)} = \frac{A_{r0}\omega L}{2 f_{resCor}\left(\phi_0\right)}}$$

Then max kinetic energy at all ϕ$_0$ was: $\boxed{K_{\max} = \tfrac{1}{2} L v_{rms}^2 \mu_m}$, which lead to

$$\boxed{K_{\max} = \tfrac{1}{2} L v_{rms}^2 \mu_m = \tfrac{1}{2} L \left(\frac{A_{r0}^2 \omega^2 L^2}{4 f_{resCor}^2}\right) \mu_m = \frac{A_{r0}^2 \omega^2 L^3}{8 f_{resCor}^2} \left(\rho_m Bh\right)}$$

($\mu_m = hB\rho_m$ is the linear mass-density)

Quality factor calculated using kinetic energy is:

$$Q=\frac{2\pi K_{\max}}{TP_{avg}}=\frac{2\pi\tfrac{1}{2}Lv_{rms}^2\mu_m}{\frac{1}{f}\tfrac{1}{2}\rho_{air}C_{DP}A_{wing}v_{rms}^3}$$

$$=\frac{L\left(\rho_m Bh\right)}{\rho_{air}C_{DP}\left(BL\right)}\frac{2\pi f}{v_{rms}}=\frac{\rho_m h}{\rho_{air}C_{DP}}\frac{\omega}{A_{r0}\omega L/\left(2f_{resCor}\right)}$$

$$=\frac{2f_{resCor}\rho_m h}{\rho_{air}C_{DP}A_{r0}L}$$

where the resonant frequency correction factor is $f_{resCor}\approx 1+2.0\text{e-}2\phi_0^2+2.4\text{e-}4\phi_0^4$ from equation (2.4), power-drag coefficient is defined as in equation (3.4), $C_{DP}=\frac{2F_P}{\rho Av_{rms}^2}=\rho\frac{2F_P}{\mu^2\,\mathrm{Re}^2}$, and is only a function of Reynolds number, defined by equation (3.2), $\mathrm{Re}\equiv\rho\frac{v_{rms}\sqrt{A}}{\mu}$ (where $\rho\,\&\,\mu$ are the air density and dynamic viscosity). The power force F_P is related to the fluid power dissipation rate per wing through the velocity, as in equation (3.3), $F_P\equiv\frac{P_A}{v_{rms}}$, and so the power dissipated into the fluid per wing is $P_A=\frac{1}{2}\rho C_{DN}Av_{rms}^3$.

As a check, we should be able to also calculate the quality factor using the potential energy:

Quality factor using potential energy: $U_{\max}=\tfrac{1}{2}EI\frac{A_r^2}{L}3.09059$, so

$$Q = \frac{2\pi U_{\max}}{TP_A} = \frac{2\pi \tfrac{1}{2} EI \dfrac{A_{r0}^2}{L} 3.09059}{\dfrac{1}{f} \tfrac{1}{2} \rho_{air} C_{DP} v_{rms}^3 A_{wing}}$$

$$= \frac{\omega EIA_{r0}^2 3.09059}{L\rho_{air} C_{DP} \left(\dfrac{A_{r0}\omega L}{2 f_{resCor}} \right)^3 A_{wing}} = \frac{EI 3.09059}{\rho_{air} C_{DP} A_{r0} \dfrac{\omega^2 L^4}{8 f_{resCor}^3} A_{wing}}$$

$$= \frac{EI 3.09059}{\rho_{air} C_{DP} A_{r0} \dfrac{\left(\dfrac{\lambda^2}{L^2} \sqrt{\dfrac{EI}{Bh\rho_m}} f_{resCor} \right)^2 L^4}{8 f_{resCor}^3} BL} = \frac{3.09059 \times 8}{\lambda^4} \frac{f_{resCor} \rho_m}{\rho_{air} C_{DP} A_{r0}} \frac{h}{L}$$

And it turns out $3.09059 = \frac{\lambda^4}{4}$, making the two formulations equivalent.

Section 4.2.4 Wing Net Force and Weight

As mentioned in Section 3.2.1, the simulations determine the net drag coefficient of the air on the wings due to flapping, C_{DN}. This drag coefficient is a function of only Reynolds number, static bending angle, and resonant bending amplitude, $C_{DN}\left(\mathrm{Re}, \phi_0, Ar_0\right)$, and determines the robot's total net force via $F_N \equiv \tfrac{1}{2} \rho_{air} C_{DN} A_{Wings} v_{rms}^2$.

Since $\mathrm{Re} = \rho_{air} \frac{v_{rms} \sqrt{A_{Wing}}}{\mu}$, another way to write this is $F_N = N_{Wings} \times \frac{1}{2\rho_{air}} C_{DN} \left(\mu \mathrm{Re}\right)^2$.

Section 4.2.4.1 Wing Lift Ratio

The wing lift ratio is the ratio of the net force to the weight of the wings:

$$r_{WL} \equiv \frac{F_{N4}}{\left(4 \times LBh\rho_m\right) g}$$

A wing lift ratio < 1 would mean the robot cannot even lift its own wings' weight. Other things equal, wings that produce much more thrust than they themselves weigh are good wings, so r_{WL} is a figure of merit.

Section 4.2.5 Power

The power available to the robot comes from two sources:

1. a volume-based power source, such as a battery or supercapacitor, and
2. an area-based solar cell power source over the wings.

In equation form, the available power is:

$$\boxed{P_{avail} = \rho_{PBatt} m_{Batt} + \sigma_{PSolar} LB \cdot N_{wings}}$$

(since chord length is $B = \frac{L}{4}$ and $N_{wings} = 4$, this is also $\boxed{P_{avail} = \rho_{PBatt} m_{Batt} + \sigma_{PSolar} L^2}$)

where:

1. m_{Batt} is the mass of the battery (or supercapacitor),
2. ρ_{PBatt} is the power density (power per volume) of the battery (or supercapacitor), and
3. σ_{PSolar} is the solar cell power density (power per area).

Since more battery power is better, other things equal, the battery mass is set at the heaviest one each robot could lift and still have excess force to move forward while tilted at a 45^o angle. This means the vertical force component is $\cos 45° = \frac{1}{\sqrt{2}}$ times the net force, and altitude is maintained when $W = \frac{1}{\sqrt{2}} F_{Net} \to m_{Total} g = \frac{1}{\sqrt{2}} F_{Net}$.

The total mass has 3 components:

1. battery,
2. wings, and
3. core (the central fuselage).

The size of the core depends on miniaturization of computation elements and other circuits. This is improving with Moore's law so it may be possible to actually ignore it. A more cautious estimate would be 4 wing's worth of mass (to maintain structural integrity for large designs) plus some fixed mass amount (to reflect the fact that some fixed space is required for computation circuitry regardless of robot size). In any case, we have that:

$$m_{Batt} = m_{Total} - m_{Wings} - m_{Core}$$

The battery power density depends on the energy density and dissipation time, with an upper limit (i.e., minimum discharge time) depending on the particular power source:

$$\rho_{PBatt}\left(t_{flight}\right) = \frac{\rho_E\left(t_{flight}\right)}{t_{flight}}$$

For the solar power density, assume operation in 20% of one sun with 12.5% cell efficiency: $\sigma_{PSolar} = 25\ \text{W/m}^2$. For the battery power density, assume 20 kW/kg, the power for the Aerogel ultracapacitor during its minimum discharge time of 16.25 s. published in [44]. These numbers are useful to get the general idea of the feasibility of this kind of flight, but are definitely not set in stone, and no doubt will improve with time.

Section 4.2.5.1 P_{Excess}: Excess Power beyond that Required for Flight

A very important figure of merit is the excess power: how much power is left over after accounting for the power required to fly: $P_{Excess} \equiv P_{Avail} - P_{required}$

Neglecting electrical power losses, $P_{required} = P_{A4}$, the average flapping power required for 4 wings.

Section 4.2.6   Stress and Strain Limits

Robot stacks need to take into account the limits to stress and strain that are acceptable to the material.   Recalling the curvature of the RS Bending mode from Section 2.3,

$$\Delta c(s) \equiv \frac{\partial^2 \hat{w}}{\partial s^2} = A_1 \frac{\lambda^2}{L^2}\left[\cosh\frac{\lambda}{L}s + \cos\frac{\lambda}{L}s + \frac{\cos\lambda + \cosh\lambda}{\sin\lambda + \sinh\lambda}\left(-\sin\frac{\lambda}{L}s - \sinh\frac{\lambda}{L}s\right)\right]$$

which in terms of $A_r = 2\frac{A_1}{L}$ is

$$\Delta c(s) \equiv A_r \frac{\lambda^2}{2L}\left[\cosh\frac{\lambda}{L}s + \cos\frac{\lambda}{L}s + \frac{\cos\lambda + \cosh\lambda}{\sin\lambda + \sinh\lambda}\left(-\sin\frac{\lambda}{L}s - \sinh\frac{\lambda}{L}s\right)\right]$$

The maximum of this curvature function from s = 0 to L is at s = 0, where

$$\Delta c(0) = \pm A_r \frac{\lambda^2}{2L} \times 2$$

Using the the stress and strain definitions from Section 2.2.6, the mechanical strain inside the cantilever is then

$$\varepsilon_{mech} = \varepsilon_0 - \varepsilon_{free} - \left(z - \bar{Z}\right)c$$
$$= \varepsilon_0 - \varepsilon_{free} - \left(z - \bar{Z}\right)\left(c_0 + \Delta c\right)$$

where $\bar{Z}$ is the z-coordinate of the neutral axis. The default curvature causes the static bending angle, $Lc_0 = \phi_0 \rightarrow c_0 = \phi_0 / L$, so

$$\varepsilon_{mechMax} = \varepsilon_0 - \varepsilon_{free} - \left(z - \bar{Z}\right)\left(\frac{\phi_0}{L} \pm A_r \frac{\lambda^2}{L}\right)$$

This equation, combined with material strain limits, establishes a maximum amplitude A_r for each stack structure, which is dependent the offset angle and resulting wing length.

How large is the resonant strain (“dynamic strain”) compared to the offset angle strain (“static strain”)? The bending portions of the strain equal when $A_r = \frac{\phi_0}{\lambda^2}$. For fixed stack structure and shape (inputs for the analytical framework), the amplitude where peak dynamic strain matches peak static strain is shown in the table below. This places a limit on the allowed amplitude depending on how far below the materials' strain limits the material stack operates at. I.e., for a symmetric stack, if a stack is designed so that the peak built-in strain after bending is 50% of the allowed strain limit, then the maximum allowed amplitude is A_{rMatch} / 2. This is because for a symmetric stack, bending sends the centre of both layers to zero strain, but inverts their junction-strain at the outside edges; i.e., bending strain is at the outside edge is twice the relative strain. Therefore, ϕ_0/L worth of curvature produces a net mechanical strain at the outside edges equal and opposite to the junction strain. Adding A_{rMatch} would double the bending strain, for a net of 4-1 = 3 times the strain at the outside edge. Adding $2A_{rMatch}$ would triple the bending strain, for a net of 6-1 = 5 times the strain at the outside edge, etc.

Table 4-4: Amplitude for equal strain from static and dynamic bending

ϕ_0 [^{o}]	A_{rMatch}
35	0.17
50	0.25
75	0.37
90	0.45
105	0.52
120	0.60
140	0.70

Depending on material limits, this can establish a severe constraint, especially for the lower bending angles. Fortunately, single crystal materials do not suffer from fatigue like the materials of a typical macroscopic cantilever beam, so the allowed strain limits for a resonating structure are significantly higher.

Section 4.2.7 Robot Velocity

In order to do anything besides hover, this type of robot needs to tilt forward. The tilting itself is accomplished by reducing the force output of the front wings relative to the back ones. The robot's max tilt angle θ from the vertical is where the vertical force component matches the robot's weight: $F_N \cos\theta = W = m_{total} g$, as illustrated in Figure 4-5.

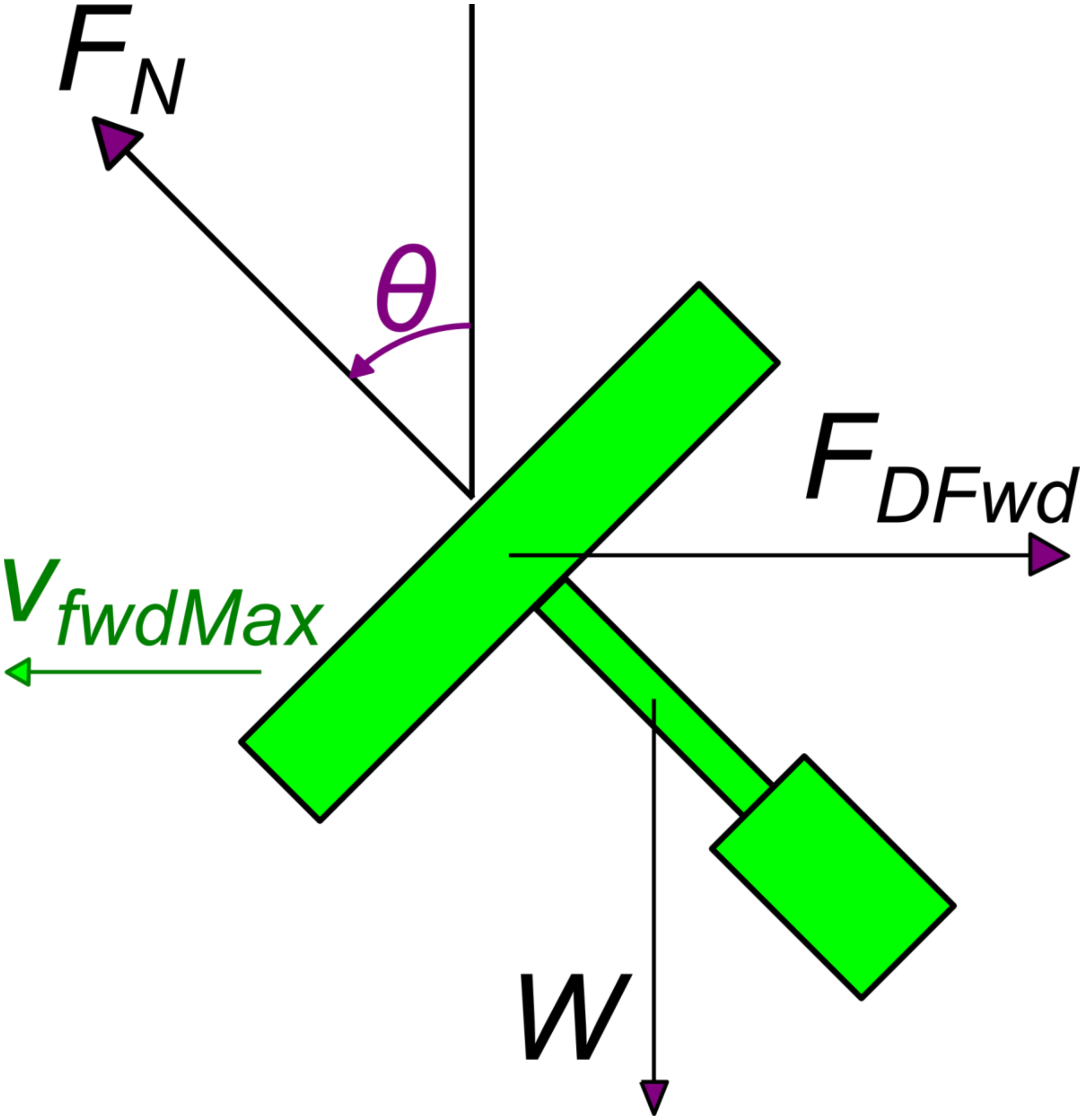


Figure 4-5: Force balance when tilting for forward velocity

The terminal forward velocity at this angle occurs when the forward force produced equals the drag force in that direction: $F_N \sin\theta = \frac{1}{2}\rho_{air} C_{Dfwd} v_{fwdMax}^2 A_{robot}$ where $A_{robot} = N_{Wing} A_{Wing} \sin\theta$ is the projected frontal area of the robot. (Note that this is only the correct area to use for sufficiently large tilts from vertical that the drag is mostly from the wing surface rather than their thickness, when skin friction would become more important).

If we select the battery so that the robot's weight is $m_{total}g = \sqrt{2}F_N$, then this conveniently sets the tilt angle to 45°. Balancing the forces for terminal forward velocity causes the angle component to drop out: $F_N \frac{1}{\sqrt{2}} = \tfrac{1}{2}\rho_{air}C_{Dfwd}v_{fwdMax}^2 N_{Wing}A_{Wing}\frac{1}{\sqrt{2}}$.

If we approximate the forward drag coefficient as $C_{Dfwd} = 0.5 + \frac{6\pi}{\mathrm{Re}_{fwd}} = 0.5 + \frac{6\pi}{\left(\frac{\rho_{air}v_{fwdMax}2L}{\mu}\right)}$, we can substitute and calculate what this terminal forward velocity is in terms of the other variables:

$$F_N = \tfrac{1}{2}\rho_{air}\left(0.5 + \mu\frac{6\pi}{\rho_{air}v_{fwdMax}2L}\right)v_{fwdMax}^2 N_{Wing}A_{Wing}$$

$$F_N = \tfrac{1}{2}\rho_{air}\left(0.5 + \mu\frac{6\pi}{\rho_{air}v_{fwdMax}2L}\right)v_{fwdMax}^2 4\frac{L^2}{4}$$

$$\frac{2}{\rho_{air}}F_N = 0.5\left(v_{fwdMax}L\right)^2 + \frac{3\pi\mu}{\rho_{air}}\left(v_{fwdMax}L\right)$$

$$0.5\left(v_{fwdMax}L\right)^2 + \frac{3\pi\mu}{\rho_{air}}\left(v_{fwdMax}L\right) - \frac{2}{\rho_{air}}F_N = 0$$

$$\rightarrow v_{fwdMax} = \frac{1}{L}\left(-\frac{3\pi\mu}{\rho_{air}} \pm \sqrt{\left(\frac{3\pi\mu}{\rho_{air}}\right)^2 + \frac{4F_N}{\rho_{air}}}\right)$$

$$v_{fwdMax} = \frac{1}{L}\frac{3\pi\mu}{\rho_{air}}\left(\pm\sqrt{1 + \frac{4\rho_{air}F_N}{(3\pi\mu)^2}} - 1\right)$$

Only the positive root makes sense since this was setup for positive v_{fwdmax}. Therefore, the maximum forward velocity in terms of the wing length and net force is:

$$\boxed{v_{fwdMax} = \frac{1}{L}\frac{3\pi\mu}{\rho_{air}}\left(\sqrt{1+\frac{4\rho_{air}F_N}{(3\pi\mu)^2}}-1\right)}$$

Section 4.2.8 Robot Range

The range is the maximum distance the robot can travel in a single sustained flight. This is also figure of merit, and is calculated from

$$range = v_{fwdMax} \times t_{flight}$$

Section 4.2.9 Electrical Properties

The electrical picture of a Quartz:Si unimorph piezoelectric cantilever when off-resonance is a voltage supply across a large capacitor in series with two small resistors. Near a resonance frequency of the cantilever, this is better modelled by the Butterworth-van Dyke (BvD) model shown in Figure 4-6 rather than only a capacitor. This must be the case because if a pure capacitor cannot dissipate electrical energy, and so there would be no way for the moving wing to do any work.

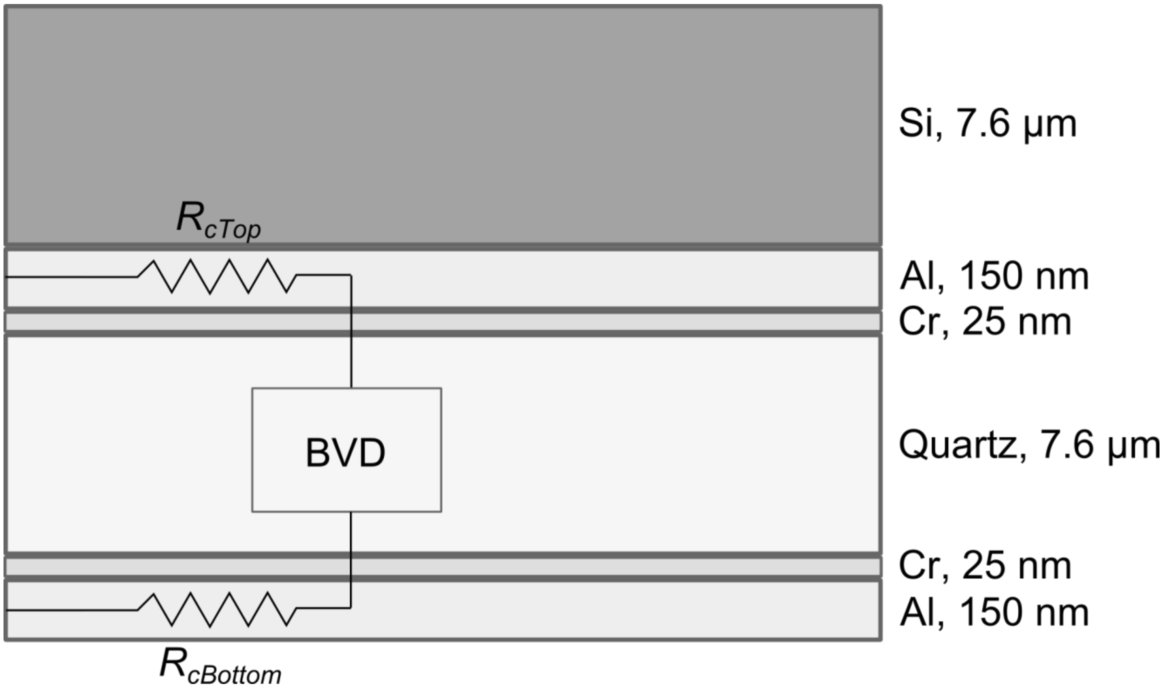


Figure 4-6: Electrical model of a stack structure resonator

The resistance of the contacts is based on an average length of *L*/2: $R_c = \rho_e \frac{l}{A} = \rho_e \frac{L/2}{Bt}$.

If instead two contact metals per side are involved, the resistance is the resistance from each in parallel: $R_{cTop} = R_{cTop1} \| R_{cTop2}$.

Since there is a contact on the top and the bottom of the wing, the total resistance in series with the piezoelectric resonator is

$R_{total} = R_{cTop} + R_{cBottom}$

Section 4.2.9.1 BvD Model: Introduction

The Butterworth-van Dyke (BvD) model for piezoelectric resonators is a capacitor (package capacitance) in parallel with an RLC branch to model the resonance of the circuit.

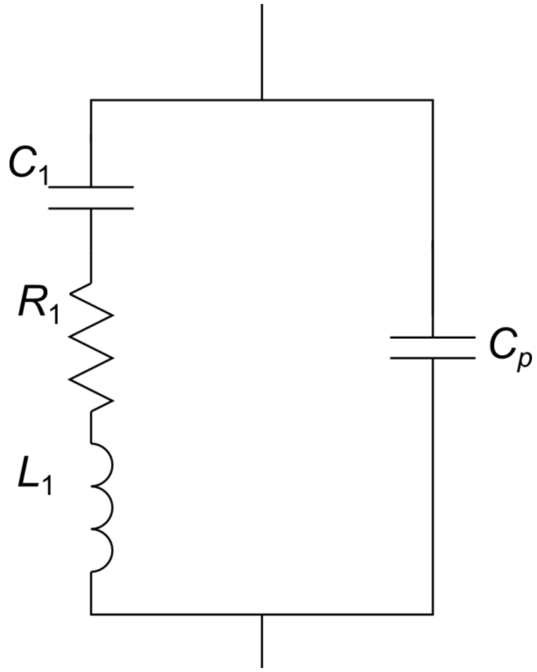


Figure 4-7: The BvD model for piezoelectric resonators

The BvD model of the resonator reflects the mechanical resonance characteristics in the electrical domain. Therefore, the energy stored *and* dissipated in the RLC branch must *equal* the energy stored and dissipated by the mechanical resonator. This allows us to calculate the wing's electrical parameters from its mechanical ones.

Section 4.2.9.2 Package Capacitance

The capacitance of a standard parallel plate capacitor is given by $C = \varepsilon \frac{A}{d} = \varepsilon_0 \varepsilon_r \frac{LB}{t}$ (that is, for 1 wing), where in this equation ε is the electrical permittivity; $\varepsilon_0 = 8.85\text{e-}12 \frac{\text{F}}{\text{m}}$ is the permittivity of free space and ε_r is the relative permittivity of the piezoelectric material in the vertical direction (for the layout of Figure 4-6). However, the coupling of mechanical and electrical domains in a piezoelectric material requires modifying this expression. From [45], "If the coupling factor is small, 0.1 or less, then [C_p] can be calculated accurately from the dielectric constant and dimensions of the resonator." The package capacitance (which is the capacitance far from resonance) for the piezoelectric layer with thickness t_i then turns out to be:

$$C_p = \frac{LB}{t_i}\left(\varepsilon_{33}^T - \frac{d_{31}^2}{s_{11}^E}\right) \quad (4.1)$$

where ε_{33} here is the permittivity in the field direction, d_{31} is the piezoelectric coupling coefficient relating the field and strain directions, and s_{11} is the compliance in the strain direction.

Section 4.2.9.3 Resonant Branch Impedance:

The RLC branch on the left side of the BvD model is known as the "resonant branch". This alone has an impedance of $R_1 + j\omega L_1 - \frac{j}{\omega C_1}$, leading to a resonance when $\omega = \omega_1$, at which point $\omega L_1 = \frac{1}{\omega C_1}$. This gives the inductance in terms of the capacitance:

$L_1 = \frac{1}{\omega_1^2 C_1}$, meaning the impedance is:

$$Z_{res} = R_1 + j\frac{1}{\omega C_1}\left(\left(\frac{\omega}{\omega_1}\right)^2 - 1\right) = R_1 + jX_r$$

Recasting this in terms of difference from resonant frequency, $\Delta\omega_1 \equiv \omega - \omega_1$, allows rewriting the impedance as

$$
\begin{aligned}
Z_{res} &= R_1 + j\frac{1}{(\Delta\omega_1+\omega_1)C_1}\left(\left(\frac{\Delta\omega_1+\omega_1}{\omega_1}\right)^2 - 1\right)\\
&= R_1 + j\frac{1}{(\Delta\omega_1+\omega_1)C_1}\left(\left(\frac{\Delta\omega_1+\omega_1}{\omega_1}\right)^2 - 1\right)\\
&= R_1 + j\frac{1}{(\Delta\omega_1+\omega_1)C_1}\left(\frac{\Delta\omega_1^2+2\omega_1\Delta\omega_1}{\omega_1^2}\right)\\
&\approx R_1 + j\frac{2\Delta\omega_1}{C_1\omega_1^2}
\end{aligned}
$$

(when sufficiently near resonance).

Section 4.2.9.4 Calculating the Resonant Branch BvD Parameters from the Mechanical Parameters

The analytical framework starts with a fixed stack structure and a fixed bending shape, amplitude, and Reynolds number. This fixes the wing dimensions, and in turn the peak mechanical energy stored in the wings $U_{\max}$, their resonant frequency ω_1, their average fluidic power dissipation P_A, and even the voltage applied to the resonant branch of the wing $v_{res}(t) = V\sin\omega t$, among other things.

We can calculate the resonant branch parameters $R_1, C_1,$ and L_1 in terms of these known values as follows:

<u>Resistance</u>

At resonance, the capacitance and inductance of the resonant branch cancel, meaning the impedance is purely resistive, $Z_{res} = R_1$. Therefore, the average power dissipated in this branch is $P_{R1} = \frac{1}{2}\frac{V^2}{R_1}$, where V is the voltage amplitude applied to the branch. Since this power dissipation is just a representation in the electrical domain of the power actually dissipated by the flapping wing, we know already that $P_{R1} = P_A$, and therefore the equivalent resistance is:

$$R_1 = \frac{V^2}{2P_A} \qquad (4.2)$$

Inductance

The resonator branch is a series circuit, and therefore has the same current everywhere. Since the impedance at resonance is R_1, we can find this current from the applied voltage:

$i_{res}(t) = \frac{V}{R_1} \sin \omega t$.

The energy stored in an inductor is given by the inductance and the current passing through it, $E_L = \frac{1}{2} Li^2$, so the max stored energy in the BvD inductor must be

$E_{LMax} = \frac{1}{2} L_1 \frac{V^2}{R_1^2}$.

Since again the RLC circuit models the resonator, this energy must be equal to the known maximum RS-mode mechanical energy $U_{\max}$, which determines the BvD inductance:

$$L_1 = \frac{2R_1^2}{V^2} U_{\max} = \frac{8P_A}{V^4} U_{\max} \qquad (4.3)$$

Capacitance

As in the previous section, for the inductance and capacitance to cancel at resonance, we need $\omega_1 L_1 = \frac{1}{\omega_1 C_1}$. Therefore the capacitance must be:

$$C_1 = \frac{1}{\omega_1^2 L_1} = \frac{V^4}{8\omega_1^2 P_A U_{\max}} \qquad (4.4)$$

Section 4.2.10 Extra Electrical Power Losses and Required Voltage

The electrical contacts on each side of the wings have nonzero resistance, and thus lead to additional power losses. In typical circuits, the wires are made thick enough that their power losses are ignored next to the device, but in the complete mechatronics design the wings require, thicker contacts can lead to appreciably more robot weight and altered resonance characteristics. As such, we are tempted to make the contacts as thin as possible, but effectively zero thickness contacts would mean that most of the power is dissipated there rather than in the wings. To determine how thin the contacts can be before this becomes a problem, we need to calculate the power losses through the contacts.

Consider a unimorph design, as shown in Figure 4-6. Let the top contact resistance be R_{cTop}, and the bottom be $R_{cBottom}$.

The complete BvD impedance is the impedance of the resonant branch (which at resonance is R_1) in parallel with the impedance of the package capacitance, $Z_p = \frac{-j}{\omega C_p}$,

$Z_{BVD} = R_1 \parallel Z_p$.

This leads to a current $I = \frac{V}{Z_{BVD}}$, where V is the voltage applied to the resonant branch (which is in parallel with the package capacitance in the BvD model, and therefore is also the voltage across the entire BvD block).

Now, this current is also the current in the contact resistors (since it is a series circuit), and so leads to an extra power loss of

$$P_{Elec} = RI_{rms}^2 = \left(R_{cTop} + R_{cBottom}\right)\frac{I^2}{2}$$

$$= \left(R_{cTop} + R_{cBottom}\right)\frac{V^2}{2\left|R_1 \parallel Z_p\right|^2}$$

The contacts also lead to a different total applied voltage than the one that actually appears on the resonator material:

$$V_{Total} = V_{cTop} + V + V_{cBottom}$$

$$= I\left(R_{cTop} + R_{cBottom}\right) + V$$

$$= \frac{V}{Z_{BVD}}\left(R_{cTop} + R_{cBottom}\right) + V$$

$$= \left(\frac{R_{cTop} + R_{cBottom}}{R_1 \parallel Z_p} + 1\right)V$$

Thus, for non-ignorable contact resistance and package capacitance, this total applied voltage is both higher and at a different phase from the voltage that appears across the resonator material.

Section 4.3 Analytical Framework Explained: Start to Finish

Each CFX simulation flaps a curved rectangle with a specific offset angle (ϕ_0), relative tip amplitude (A_{r0}), and Reynolds number (Re). The wings always have chord length equal to one-quarter their wing length ($B = L/4$). The simulations output the instantaneous force and power dissipation which I average over the final few cycles (once the flow has more-or-less stabilized) to calculate the net force (F_N) and average power dissipated in the fluid (P_A), respectively.

For fixed fluid viscosity and density, the net force is invariant with L and flapping frequency (f) at constant Re & shape (ϕ_0 & A_{r0}), but the dissipated power is not. To get around this we can use the related quantity which *is* invariant at constant Re & shape (for a fixed fluid), the *power force,* $F_P \equiv P_A / v_{rms}$ where $v_{rms}\left(\phi_0\right) = \frac{v_{rms}\left(0\right)}{f_{resCor}\left(\phi_0\right)} = \frac{A_{r0}\omega L}{2 f_{resCor}\left(\phi_0\right)}$.

Further, defining the net and power drag coefficients as $C_{DN} \equiv \frac{F_{Net}}{\frac{1}{2}\rho v_{rms}^2 A_{wing}}$ and $C_{DP} \equiv \frac{F_P}{\frac{1}{2}\rho v_{rms}^2 A_{wing}} \equiv \frac{P_A}{\frac{1}{2}\rho v_{rms}^3 A_{wing}}$ gives quantities which are only functions of Reynolds number and shape, even across different fluid densities and viscosities. These then define the force and power for a given wing:

$$F_{Net}\left(L, A_{r0}, \phi_0, f, \rho_{air}, \mathrm{Re}\right) = \tfrac{1}{2}\rho_{air} C_{DN}\left(\mathrm{Re}, A_{r0}, \phi_0\right) \times v_{rms}^2 A_{wings}$$
$$P_A\left(L, A_{r0}, \phi_0, f, \rho_{air}, \mathrm{Re}\right) = \tfrac{1}{2}\rho_{air} C_{DP}\left(\mathrm{Re}, A_{r0}, \phi_0\right) \times v_{rms}^3 A_{wings}$$

Next, we select a piezoelectric "stack structure" to make up the wings. This consists of picking the order and *relative* thickness (that is, thickness of each layer as a percentage of the total wing thickness (*h*), but *not* the thickness itself) of all the layers from bottom to top that make up the wings, along with their material parameters and built-in strains after fabrication.

The stack structure and shape then uniquely lead to a final robot design via the analytical framework, displayed in detail in Figure 4-8.

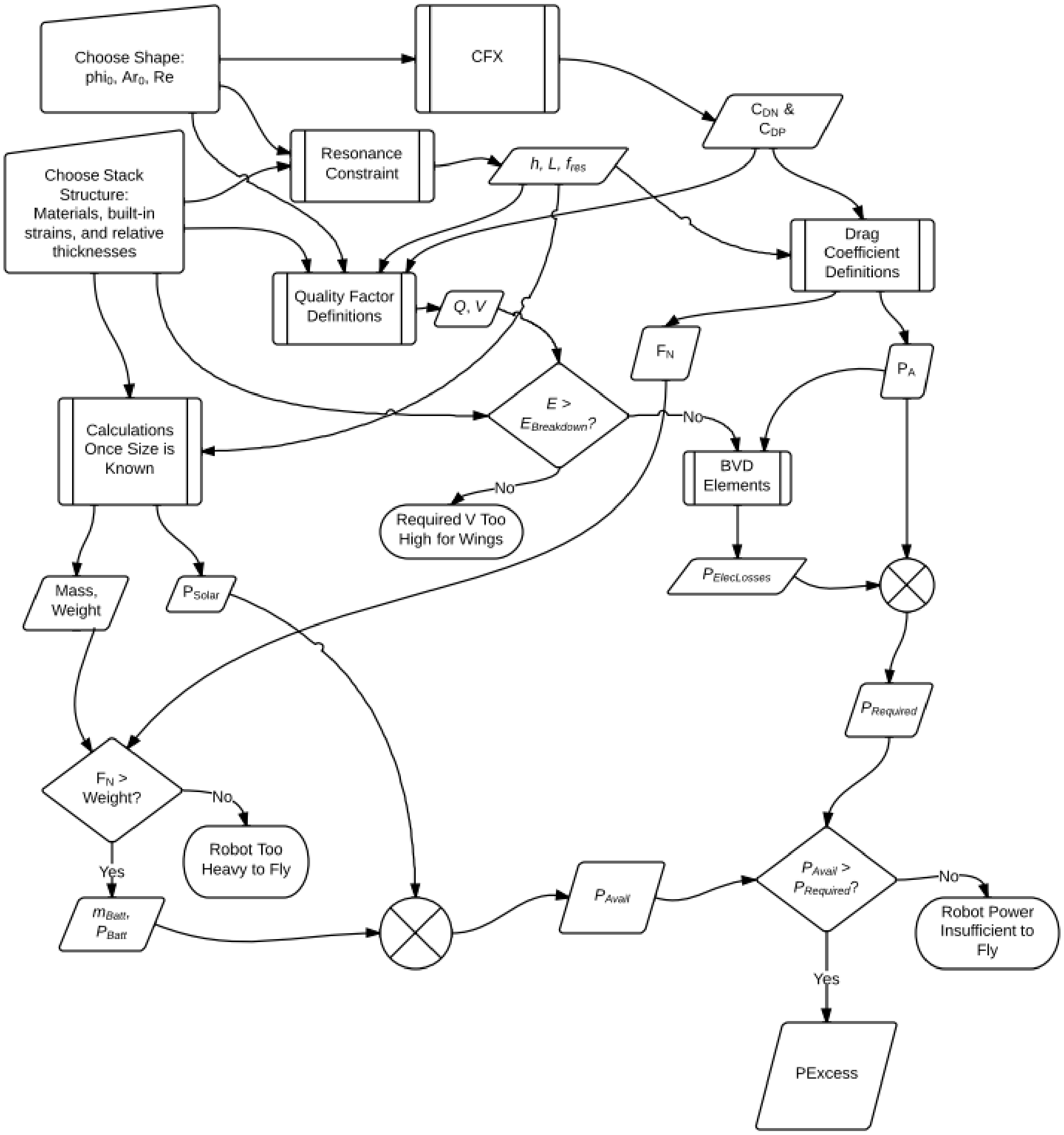

Choose Shape: phi_0, Ar_0, Re
CFX
C_{DN} & C_{DP}
Resonance Constraint
h, L, f_{res}
Choose Stack Structure: Materials, built-in strains, and relative thicknesses
Drag Coefficient Definitions
Quality Factor Definitions
Q, V
F_N
P_A
$E > E_{Breakdown}$?
No
Calculations Once Size is Known
BVD Elements
No
Required V Too High for Wings
$P_{ElecLosses}$
Mass, Weight
P_{Solar}
$P_{Required}$
F_N > Weight?
No
Robot Too Heavy to Fly
Yes
m_{Batt}, P_{Batt}
P_{Avail}
$P_{Avail} > P_{Required}$?
No
Robot Power Insufficient to Fly
Yes
PExcess

Figure 4-8: Detailed flowchart for the analytical framework

The details of the steps in this figure are sketched out in the following sections. Following that, we will apply the routine to some wing designs and determine whether the force and power are sufficient for some robots to fly, and if so, which ones fly best.

Section 4.3.1 Analytical Framework Discussion, Based on Equation Sections

Section 4.3.1.1 Resonance Constraint

The stack structure choice or CFX simulation inputs alone do not uniquely determine the wing length, total thickness, or resonant frequency. However, when combining them at resonance we obtain three equations that do combine to determine *L*, *h*, and *f*:

1. Reynolds number of the wings
2. Initial bending angle of the wings
3. Resonant frequency of the wings

1. The CFX simulations are run at a specific Reynolds number. Since the operating air density and viscosity of the final robot are constant (at STP values for air), a fixed Reynolds number relates the wing length and frequency:

$$\mathrm{Re} \equiv \rho_{air} \frac{v_{rms}\sqrt{A}}{\mu} = \rho_{air} \frac{\left(\dfrac{\pi A_{r0} f L}{f_{resCor}}\right)\sqrt{\dfrac{1}{4}L^2}}{\mu}$$

$$\mathrm{Re} = \frac{\rho_{air}}{\mu} \frac{\pi}{2} \frac{A_{r0}}{f_{resCor}} f L^2$$

Therefore $\boxed{f = \dfrac{\mu}{\rho_{air}} \dfrac{2}{\pi} \dfrac{\mathrm{Re}\, f_{resCor}}{A_{r0}} \dfrac{1}{L^2}}$

2. The CFX simulations are also run at a specific built-in bending angle ϕ_0. For a given stack structure, fixing the bending angle ϕ_0 relates the wing thickness to the wing length:

Bending angle is $\phi_0 = c_0 L$, where the built-in curvature is

$$c_0 = \frac{1}{1 - \dfrac{1}{2}\mathbf{DA}^{-1}\mathbf{B}} \mathbf{DA}^{-1}\mathbf{C}_0 = \gamma_{c0} \frac{1}{h}\text{; therefore: } \boxed{\phi_0 = \gamma_{c0} \frac{L}{h}}$$

3. The wing is operating at its resonant frequency:

$$\boxed{f = \frac{\lambda^2}{2\pi}\sqrt{\frac{E_Y}{12\rho_m}}\, f_{resCor}\frac{h}{L^2}}$$

Again, $\frac{E_Y}{\rho_m}$ depends only on the stack structure (density is a thickness-weighted average), and f_{resCor} depends only on the initial bending angle (i.e., the CFX simulation chosen), so with a fixed CFX simulation and stack structure, everything here is fixed except for *h* & *L*.

Thus, we've arrived at three independent equations in the three unknowns *h*, *L*, and *f*:

1. the Reynolds number of the wing matches that of the simulation:

   $$f = \frac{\mu}{\rho_{air}}\frac{2}{\pi}\frac{\mathrm{Re}\, f_{resCor}}{A_{r0}}\frac{1}{L^2},$$

2. the bending angle of the wing matches that of the simulation: $\phi_0 = \gamma_{c0}\frac{L}{h}$, and
3. the frequency of the wing matches its resonant frequency:

   $$f = \frac{\lambda^2}{2\pi}\sqrt{\frac{E_Y}{12\rho_m}}\, f_{resCor}\frac{h}{L^2}$$

Solve them by direct substitution:

1. *Combine the resonant frequency equation with the Reynolds number definition; fixes h for given stack, air, and shape:*

$$f_{res} = f$$

$$\frac{\lambda^2}{2\pi}\sqrt{\frac{E_Y}{12\rho_m}} f_{resCor}\frac{h}{L^2} = \frac{\mu}{\rho_{air}}\frac{2}{\pi}\frac{\mathrm{Re}\, f_{resCor}}{A_{r0}}\frac{1}{L^2}$$

$$\lambda^2\sqrt{\frac{E_Y}{12\rho_m}}\cdot h = \frac{\mu}{\rho_{air}}4\frac{\mathrm{Re}}{A_{r0}}$$

$$\therefore h = \frac{\mu}{\rho_{air}}\frac{4\,\mathrm{Re}}{A_{r0}\lambda^2}\sqrt{\frac{12\rho_m}{E_Y}}$$

Note: *h* depends on the stack's ratio of effective density to Young's modulus, and the simulation's ratio of Reynolds number to amplitude. However, *h* is independent of the offset angle and the stack's curvature constant.

2. *Known h and stack then determine L from the resting angle:*

$$\phi_0 = \gamma_{c0}\frac{L}{h}$$

$$\to L = \frac{h\phi_0}{\gamma_{c0}} = \frac{\phi_0}{\gamma_{c0}}\frac{\mu}{\rho_{air}}\frac{4\,\mathrm{Re}}{A_{r0}\lambda^2}\sqrt{\frac{12\rho_m}{E_Y}}$$

3. *Finally, backsubstitute h & L to determine the resonant frequency for the stack and shape:*

$$
\begin{aligned}
f &= \frac{\lambda^2}{2\pi}\sqrt{\frac{E_Y}{12\rho_m}}\, f_{resCor}\,\frac{h}{L^2} \\
&= \frac{\lambda^2}{2\pi}\sqrt{\frac{E_Y}{12\rho_m}}\, f_{resCor}\,\frac{\left(\dfrac{\mu}{\rho_{air}}\dfrac{4\,\mathrm{Re}}{A_{r0}\lambda^2}\sqrt{\dfrac{12\rho_m}{E_Y}}\right)}{\left(\dfrac{\phi_0}{\gamma_{c0}}\dfrac{\mu}{\rho_{air}}\dfrac{4\,\mathrm{Re}}{A_{r0}\lambda^2}\sqrt{\dfrac{12\rho_m}{E_Y}}\right)^2} \\
&= \frac{\lambda^2}{2\pi}\sqrt{\frac{E_Y}{12\rho_m}}\, f_{resCor}\,\frac{1}{\dfrac{\phi_0^2}{\gamma_{c0}^2}\left(\dfrac{\mu}{\rho_{air}}\dfrac{4\,\mathrm{Re}}{A_{r0}\lambda^2}\sqrt{\dfrac{12\rho_m}{E_Y}}\right)} \\
&= \frac{\lambda^2}{2\pi}\frac{E_Y}{12\rho_m}\, f_{resCor}\,\frac{1}{\dfrac{\phi_0^2}{\gamma_{c0}^2}\left(\dfrac{\mu}{\rho_{air}}\dfrac{4\,\mathrm{Re}}{A_{r0}\lambda^2}\right)} \\
&= \frac{\lambda^4}{2\pi}\frac{E_Y}{12\rho_m}\, f_{resCor}\,\frac{\gamma_{c0}^2}{\phi_0^2}\frac{\rho_{air}}{\mu}\frac{A_{r0}}{4\,\mathrm{Re}} \\
&= \frac{\lambda^4}{2\pi}\frac{\rho_{air}}{\mu}\frac{E_Y\gamma_{c0}^2}{48\rho_m}\frac{f_{resCor}A_{r0}}{\mathrm{Re}}
\end{aligned}
$$

Section 4.3.1.2 Stress Limit Constraint

The stack structure defines the built-in strain of each layer before bending. Over most of the wing, this is relieved during bending but can be locally enhanced near the material interfaces. Resonanting further increases the local strain proportional to the amplitude through the curvature: $\Delta\varepsilon_{bending} =$

Section 4.3.1.3 Quality Factor Definitions

Next, appeal to the dual definitions of the quality factor as amplitude ratio and energy ratio.

Quality factor as an amplitude ratio:

We know that $A_{r0DC} = 0.445\Phi_{DC}$, where DC bending angle is

$\Phi_{DC} \equiv c_{piezo}L = \left(\gamma_{cpiezo}\frac{V}{h^2}\right)L$ so define $A_{r0DC} \equiv \gamma_{ArDC}\frac{LV}{h^2}$ where $\gamma_{ArDC} = 0.445\gamma_{cpiezo}$.

Then the quality factor is the ratio of the resonant amplitude to this amplitude:

$$Q = \frac{A_{r0}}{A_{r0DC}}$$

$$Q = \frac{A_{r0}}{\gamma_{ArDC}\frac{LV}{h^2}}$$

Quality factor as an energy ratio:

Quality factor is also the ratio of 2π times the peak energy stored to energy dissipated per cycle: $Q \equiv \frac{2\pi K_{\max}}{TP_A} = \frac{2f_{resCor}\rho_m}{\rho_{air}C_{DP}A_{r0}}$

Equating:

Equating these definitions determines the voltage amplitude applied to the resonator material:

$$\rightarrow V = \frac{h^2}{L\gamma_{ArDC}Q}A_{r0}$$

Section 4.3.1.4 Breakdown Field Check

Knowing the voltage and the thickness of the piezoelectric allows finding the electric field across the piezoelectric element: $E = \frac{-V}{t}$.

A robot could only work if this field is sufficiently lower than the breakdown field of the piezoelectric.

Section 4.3.1.5 Drag Coefficient Definitions:

With the size and shape determined, we can substitute the h, L, & f_{res} into the equations for the drag coefficients to determine the net force and power used by flapping:

$$F_{Net}\left(L, A_{r0}, \phi_0, f, \rho_{air}, \mathrm{Re}\right) = \tfrac{1}{2}\rho_{air} C_{DN}\left(\mathrm{Re}, A_{r0}, \phi_0\right) \times v_{rms}^2 A_{wings}$$

$$P_A\left(L, A_{r0}, \phi_0, f, \rho_{air}, \mathrm{Re}\right) = \tfrac{1}{2}\rho_{air} C_{DP}\left(\mathrm{Re}, A_{r0}, \phi_0\right) \times v_{rms}^3 A_{wings}$$

Section 4.3.1.6 BvD Elements and Electrical Power Losses

With the dissipated power and dimensions of the stack now known, we can find the C_p, R_1, C_1, and L_1 of the BvD model for the electrical picture of the piezoelectric layer, and also the resistances of the top and bottom contacts to determine the electrical power losses P_{elec}.

Section 4.3.1.7 Calculations Once Size is Known

Knowing the size of the wings allows determining their mass and weight, as well as the mass and weight of the fuselage (minus the battery), and the solar power available.

Section 4.3.1.8 Net Force Check

To fly, the force output by the wings must exceed the robot weight while in forward flight, so exclude robots whose net force is not greater than the one needed to lift the wings and fuselage when tilted at 45° to the vertical: $F_N > \sqrt{2}g\left(m_{wings} + m_{core}\right)$

Section 4.3.1.9 Battery Mass and P$_{Batt}$

The battery mass is max liftable when tilted at 45° to the vertical:

$$m_{batt} = \frac{F_N}{\sqrt{2}g} - m_{wings} - m_{core}$$

With this known, the power output by the battery is $P_{batt} = m_{batt} \times \rho_{pbatt}$. This depends on the flight time chosen, t_{flight}: $\rho_{pbatt} = \begin{cases} \dfrac{\rho_{Ebatt}}{t_{flight}}, & t_{flight} > t_{flightMin} \\ \rho_{pbattMax}, & t_{flight} \le t_{flightMin} \end{cases}$.

Section 4.3.1.10 Total Available Power

The power available to the robot is from both the volume and the area power source: battery or supercapacitor combined with solar cells:

$$P_{avail} = P_{batt} + P_{solar}$$

Section 4.3.1.11 Total Required Power

The total required power for flight is the power dissipated by flapping plus the extra electrical power losses:

$$P_{required} = P_A + P_{ElecLosses}$$

Section 4.3.1.12 Power Check and Calculation of Excess Power:

The "excess power" is defined as the leftover power available from all sources after accounting for the power required to fly:

$$P_{excess} \equiv P_{avail} - P_{required}$$

This power is necessary to operate all other things on the robot, such as control, sensing, and communications circuitry. As such, $P_{excess} > 0$ is not only the final check of whether flight is possible, but also a figure of merit to be optimized in its own right.

Section 4.3.2 Figures of Merit

Is higher excess power better if it means the robot takes longer to recharge its battery? Is maximum top-speed more important than maximum average speed over a long distance? Is a bigger, faster, more powerful robot a superior choice to using the same material and fabrication time to produce many smaller robots? Selecting which robot is best out of several flying designs is not at all obvious, and depends on the relative weights of several figures of merit such as the following:

1. Max payload: excess F$_N$ at hover = $W_{Payload} = F_N - gm_{total}$
   a. Under the $m_{batt} = \frac{F_N}{\sqrt{2}g} - m_{wings} - m_{core}$ constraint, this is equal to:
   b. $$\begin{aligned} W_{Payload} &= F_N - g\left(m_{batt} + m_{wings} + m_{core}\right) \\ &= F_N - g\left(\frac{F_N}{\sqrt{2}g}\right) = F_N\left(1 - \frac{1}{\sqrt{2}}\right) \end{aligned}$$
2. Max forward velocity: $v_{fwdMax} = \frac{1}{L}\frac{3\pi\mu}{\rho_{air}}\left(\sqrt{1 + \frac{4\rho_{air}F_N}{\left(3\pi\mu\right)^2}} - 1\right)$
3. Excess power P_{excess} (the power the robot has available beyond that needed for sustained flight)
4. Operation time ratio, defined as the fraction of time the robot can spend flying:
   $OTR \equiv \frac{t_{flight}}{t_{flight} + t_{charge}}$, where $t_{charge} \equiv \frac{E_{Batt}}{P_{Solar}}$
5. Manoeuvrability: Max acceleration the robot can generate ($a_{Max} = F_N / m_{robot}$)
6. Inverse mass: 1/m_{Total}. Other things equal, lower mass means less materials are required and therefore lower cost per robot
7. Inverse WaferAreaRequired: $\frac{1}{\text{WaferAreaRequired}} = \frac{1}{\left(2L + B\right)\left(3B\right)}$. More area of a starting wafer taken up means less robots can be made on each wafer

One approach is to define a function made up of the product of all of these quantities and optimize that:

$$Merit \equiv \left(W_{Payload}\right)\left(v_{FwdMax}\right)\left(P_{Excess}\right)\left(OTR\right)\left(a_{Max}\right)\frac{1}{m_{Total}}\frac{1}{\left(2L + B\right)\left(3B\right)}$$

Priorities can be changed by adjusting the weights of each quantity as-needed. Doing so does not change the robot that each stack and shape leads to; that is fixed by the constraints of the analytical framework. Rather, choosing a merit function only affects the decision of which stack and shape we decide is best.

Section 4.4 Optimization Routine Implementation

This section walks through Phase 1 of the Optimization Routine (optimizing CFX inputs) by carrying out the Analytical Framework for all CFX inputs using the Q:Si stack structure from Section 4.2.1.4. This stack structure had the following stack parameters, shown in Table 4-5.

Table 4-5: Q:Si stack parameters

Stack	γ_{c0}	E_Y	ρ_m	γ_{cpiezo}
	µm-rad /mm	GPa	kg/m^3	µm^{2}-mrad /mm-V
Q:Si	3.446	110.8	2631	6.397

Section 4.4.1 Resonance Constraint: *h*, *L*, and *f*

Asserting that the structure operates at resonance uniquely defines the thickness, length, and frequency of every input flapping shape, given the stack parameters. The results for the Q:Si stack are shown in Figure 4-9 through Figure 4-11.

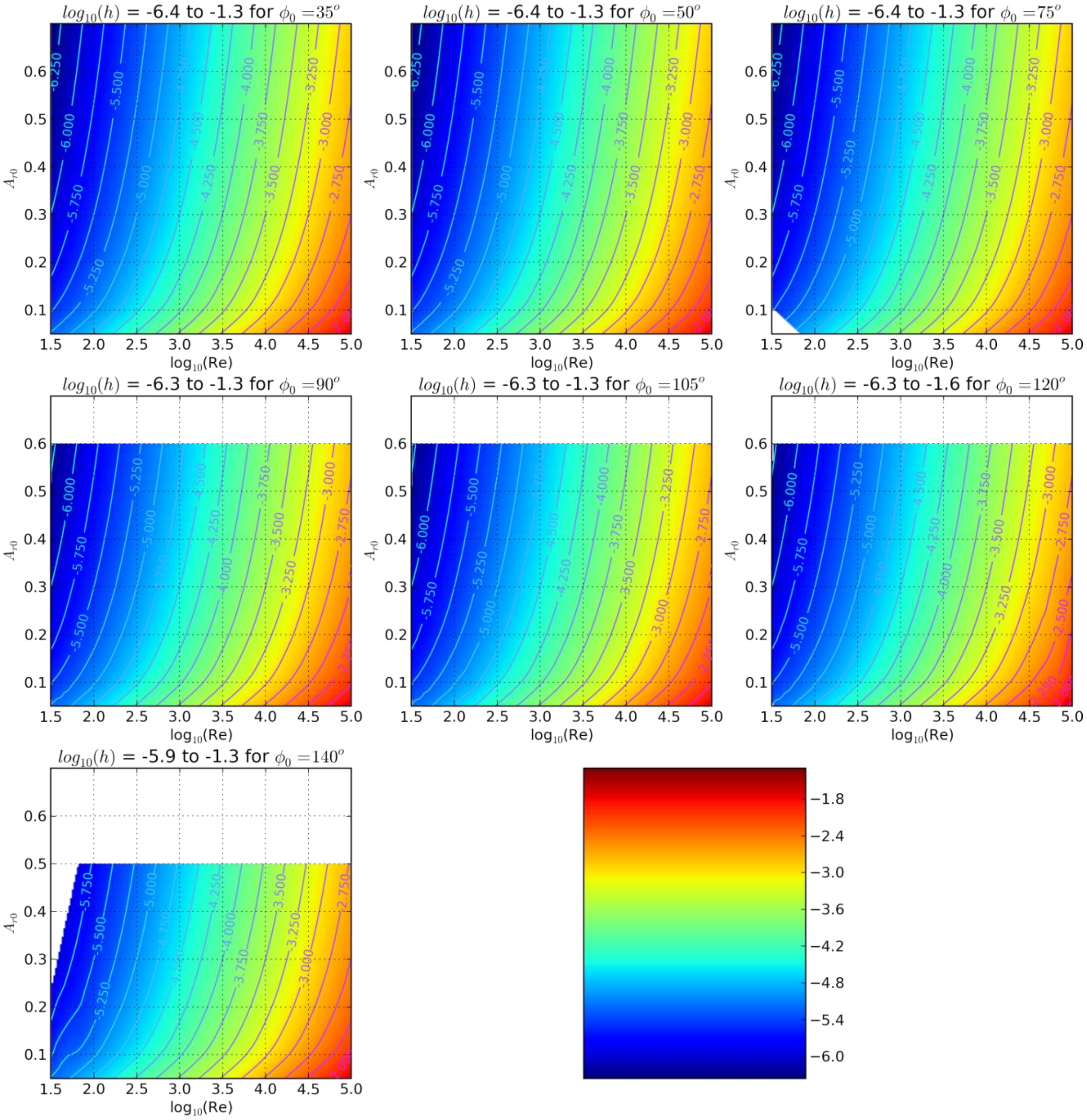


Figure 4-9: $\log_{10}$ of the wing thickness in m

The wing thickness required increases with higher Reynolds number, and decreases with increasing amplitude.

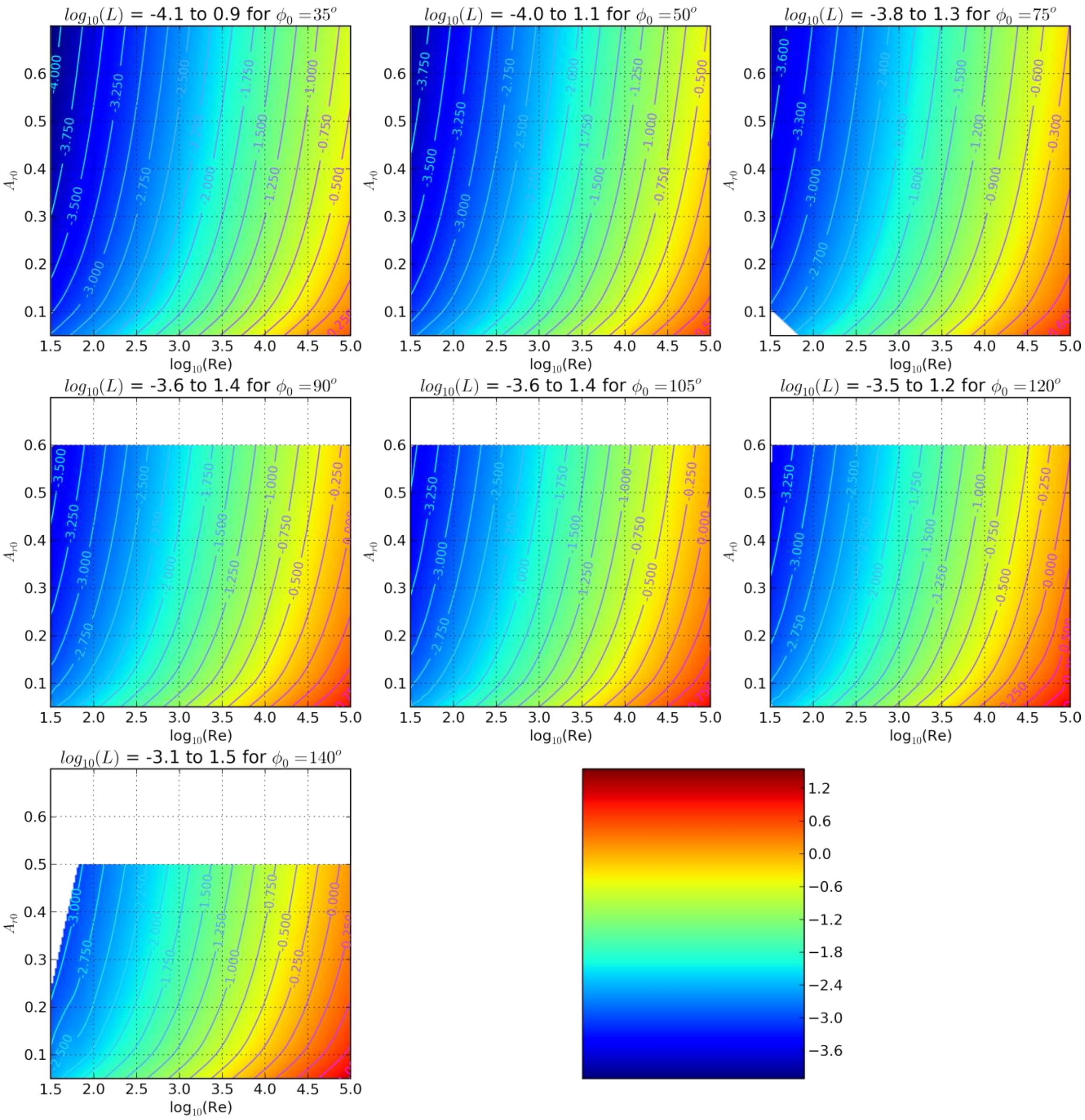


Figure 4-10: $\log_{10}$ of the wing length in m

Higher bending angles for the same stack mean larger lengths are required. Length increases with thickness and bending angle, and decreases with amplitude.

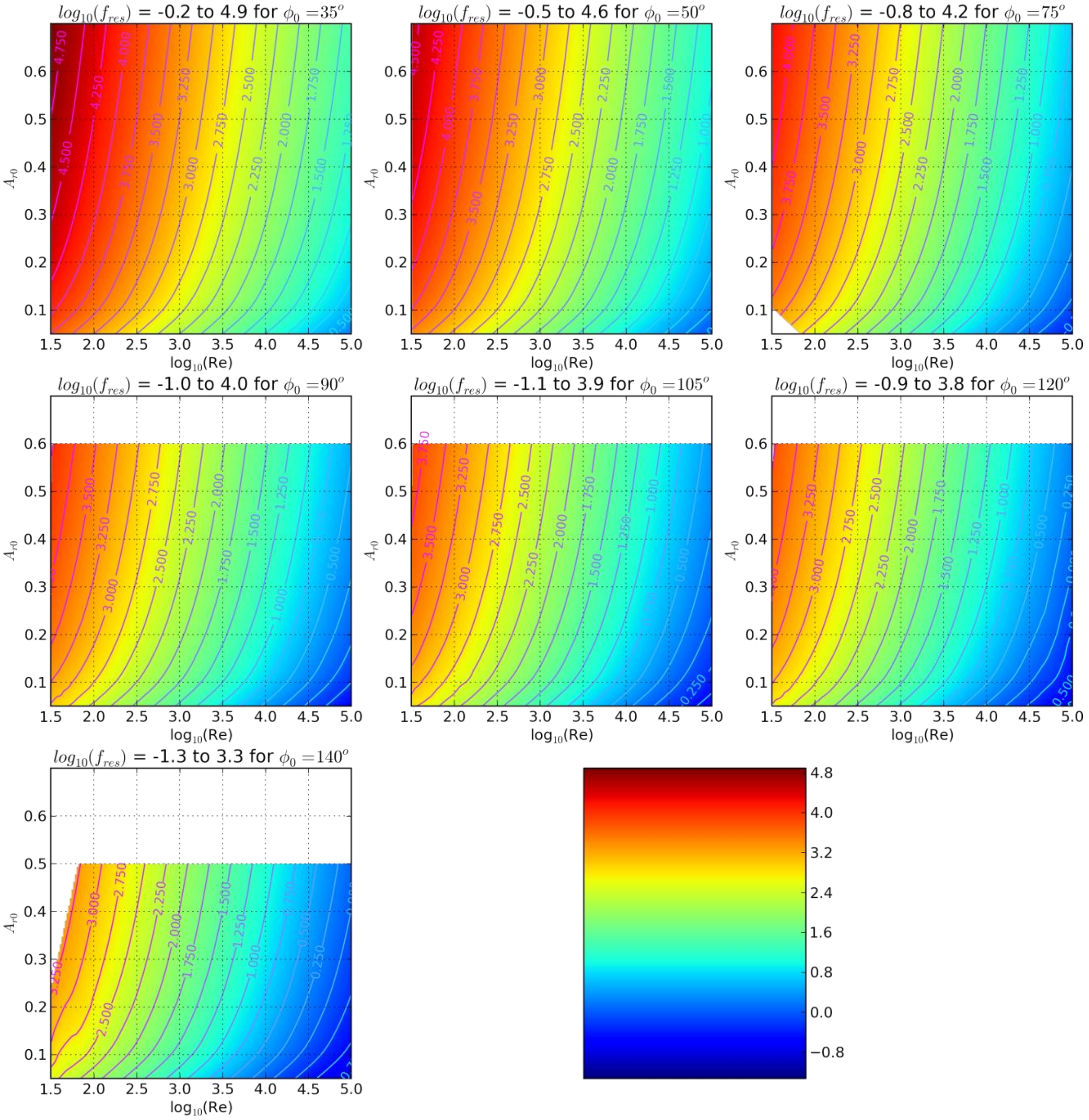


Figure 4-11: $\log_{10}$ of frequency in Hz

The frequency is inversely proportional to the length squared and proportional to the thickness. This, combined with its shape dependencies lead to frequency decreasing with Reynolds number, increasing with amplitude, and decreasing with initial bending angle.

Section 4.4.2 Wing Mass and Force

Knowing the dimensions and density we can determine the wing mass and weight to compare to its net force output. The wing weight ratio (WWR), equal to the weight of the wings divided by their net force output, does just that and is plotted in Figure 4-12. Referring to the colourbar, wings whose flapping produces enough force to lift their own weight are those whose colour is at least as blue as cyan. This requires a sufficiently high amplitude and sufficiently low Reynolds number, with the higher initial bending angle designs performing worse at this metric. This is because although the net drag coefficient (and therefore net force) at high-Re and high-A_r is better for ϕ_0 in the range of 90^o than 35^o, the 90^o bending angles require longer wings to achieve this angle for the same stack's γ_{c0}.

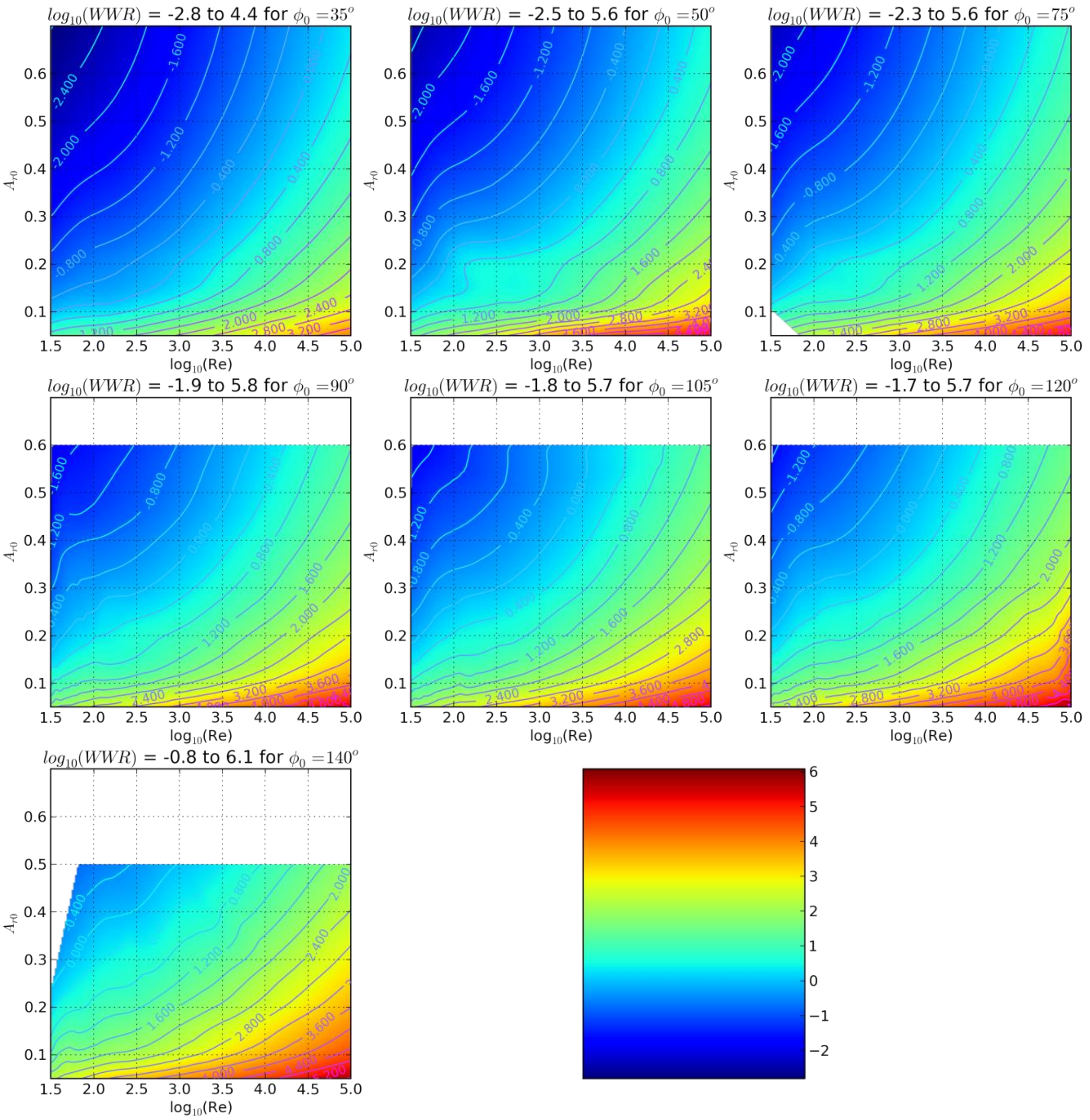


Figure 4-12: $\log_{10}$ of the wing weight ratio. Only wings with WWR < 1 ($\log_{10}$(WWR)<0) produce enough force to lift their own weight. For fixed stack curvature constant, higher bending angles require more length and consequently extra mass than they make up for with drag coefficient.

Section 4.4.3 Q, V, and *E*

Knowing the dimensions of the stack and its resonant amplitude and drag coefficients enables calculating the peak energy stored and quality factor.

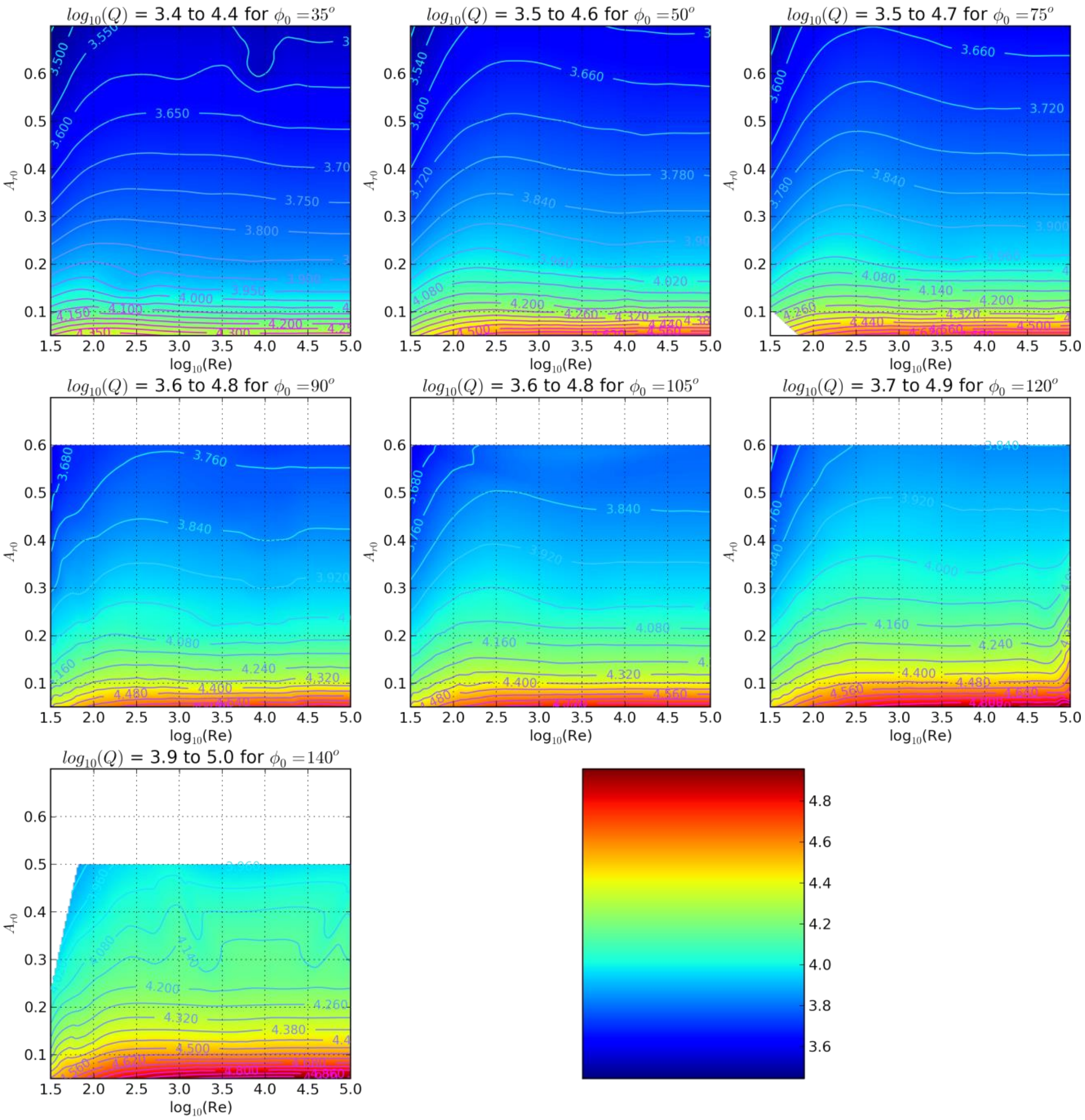


Figure 4-13: Quality factor

The quality factor increased most markedly as amplitude decreased. Although lower amplitudes lead to higher power drag coefficients and thus more dissipated energy for each Reynolds number and velocity, this was not enough to overcome the increase in their stored energy. This also increased the quality factor for higher bending angles. Interestingly, because many of the same parameters are in the numerator and denominator, the quality factor did not vary as much as many of the other parameters, spanning less than 2 orders of magnitude over the entire simulation set.

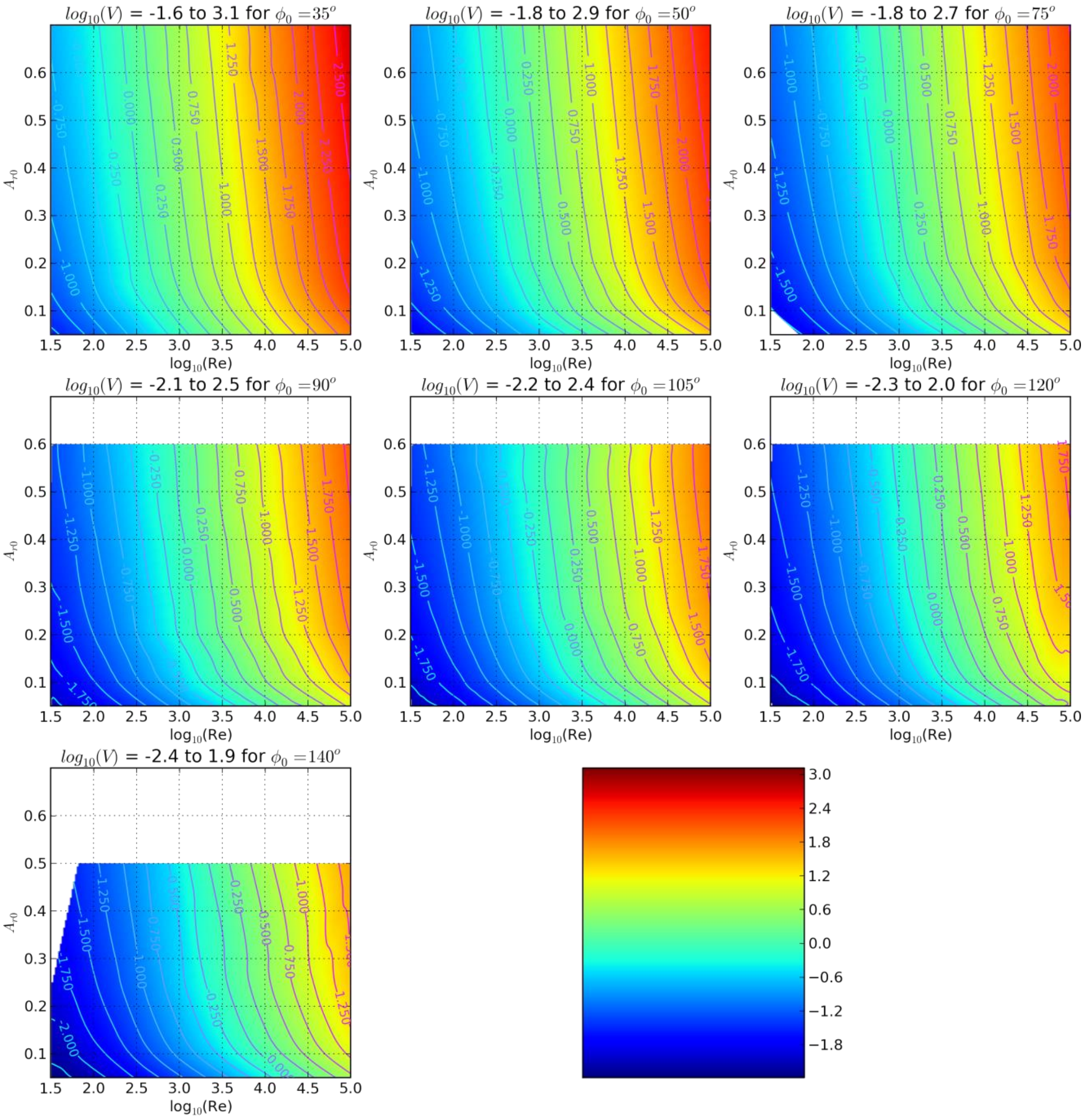


Figure 4-14: log$_{10}$ of the voltage amplitude (in V) required for to produce the wing's flapping amplitude given its quality factor

Other things equal, a higher quality factor means less required voltage to reach each bending amplitude. Higher amplitudes did lead to more required voltage, but not so much as higher Reynolds numbers. This is because as shown by Figure 4-9, higher Reynolds numbers require significantly higher thicknesses, and the electric field produced by a given voltage is inversely proportional to the thickness.

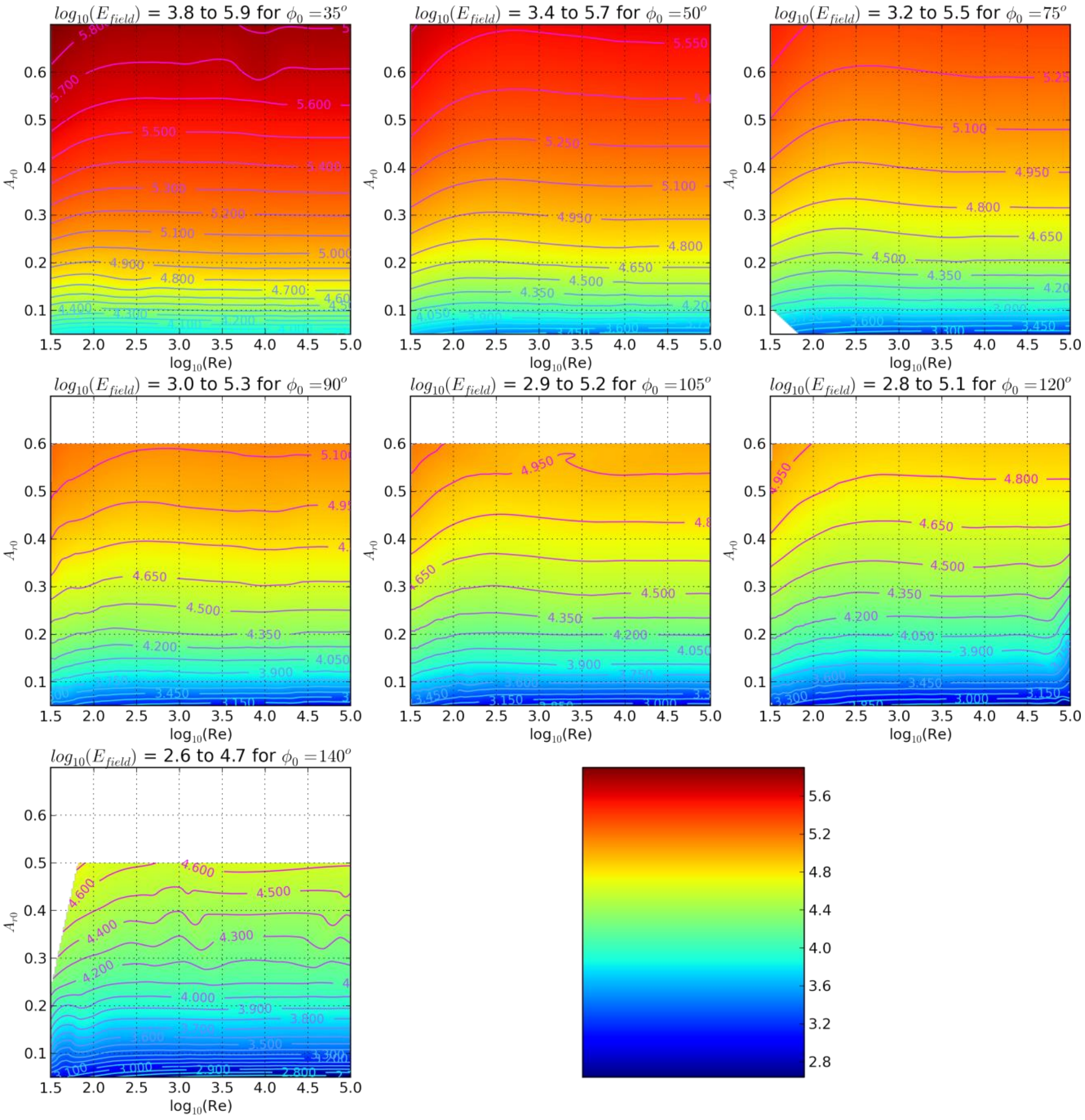


Figure 4-15: $\log_{10}$ of electric field (in N/C) required to produce the flapping amplitude

Required electric field increases with decreasing quality factor and increasing amplitude, leading to significantly higher required fields for the higher amplitudes. The breakdown field for quartz is in excess of 1 GN/C, so even using the air breakdown of approximately 40 MN/C, this constraint is not relevant for the Q:Si wing.

Section 4.4.4   Battery Mass and Powers

Restricting to the designs which can lift a battery at least as heavy as their wings means restricting to wing weight ratios lower than 0.5 (neglecting the core mass for the time being). At time of writing, the $\phi_0 = 140^o$ wings do not have enough working points to generate a contour map, and so resorted to a scatter plot.

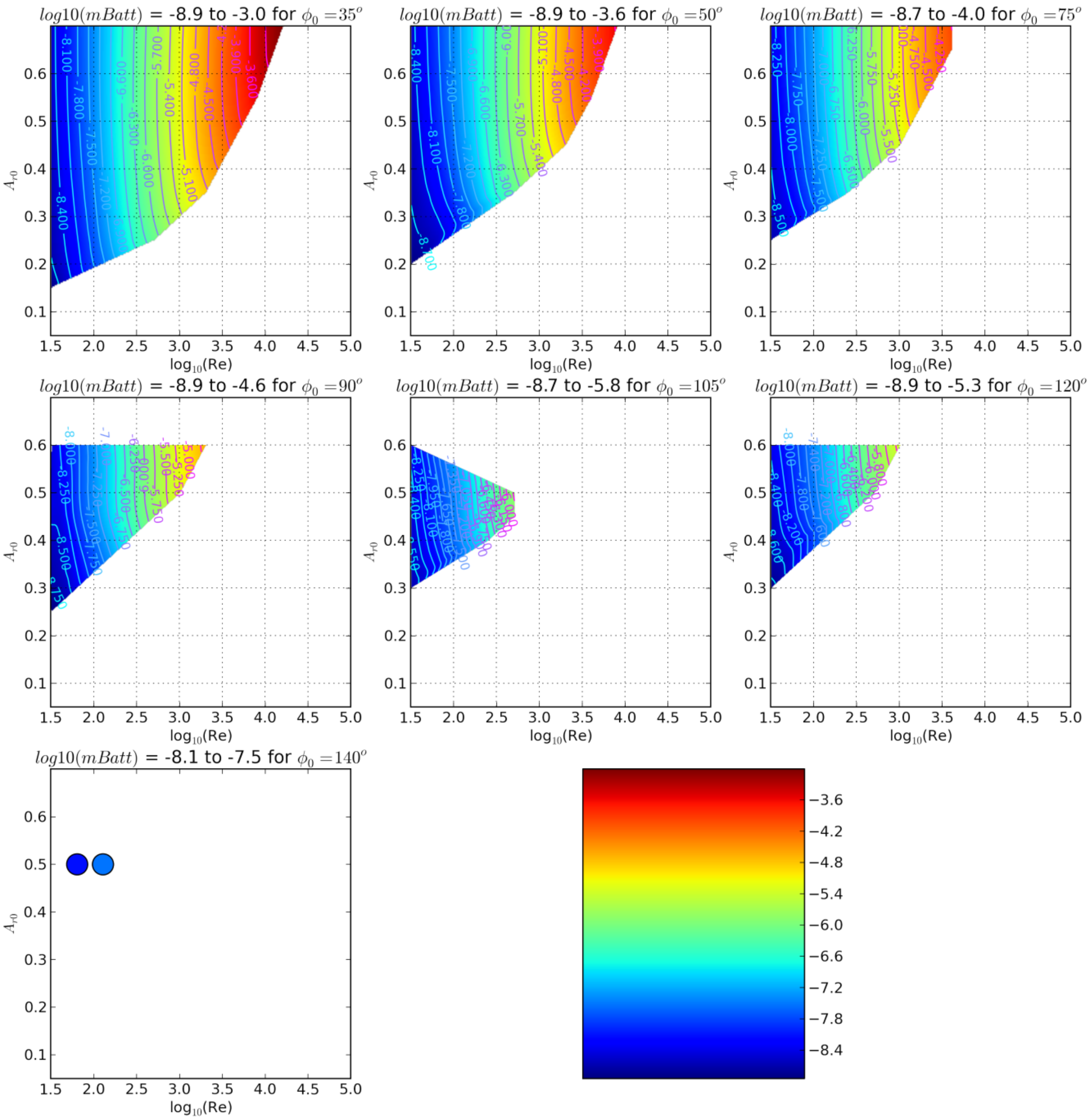


Figure 4-16: $\log_{10}$ of battery mass, in kg

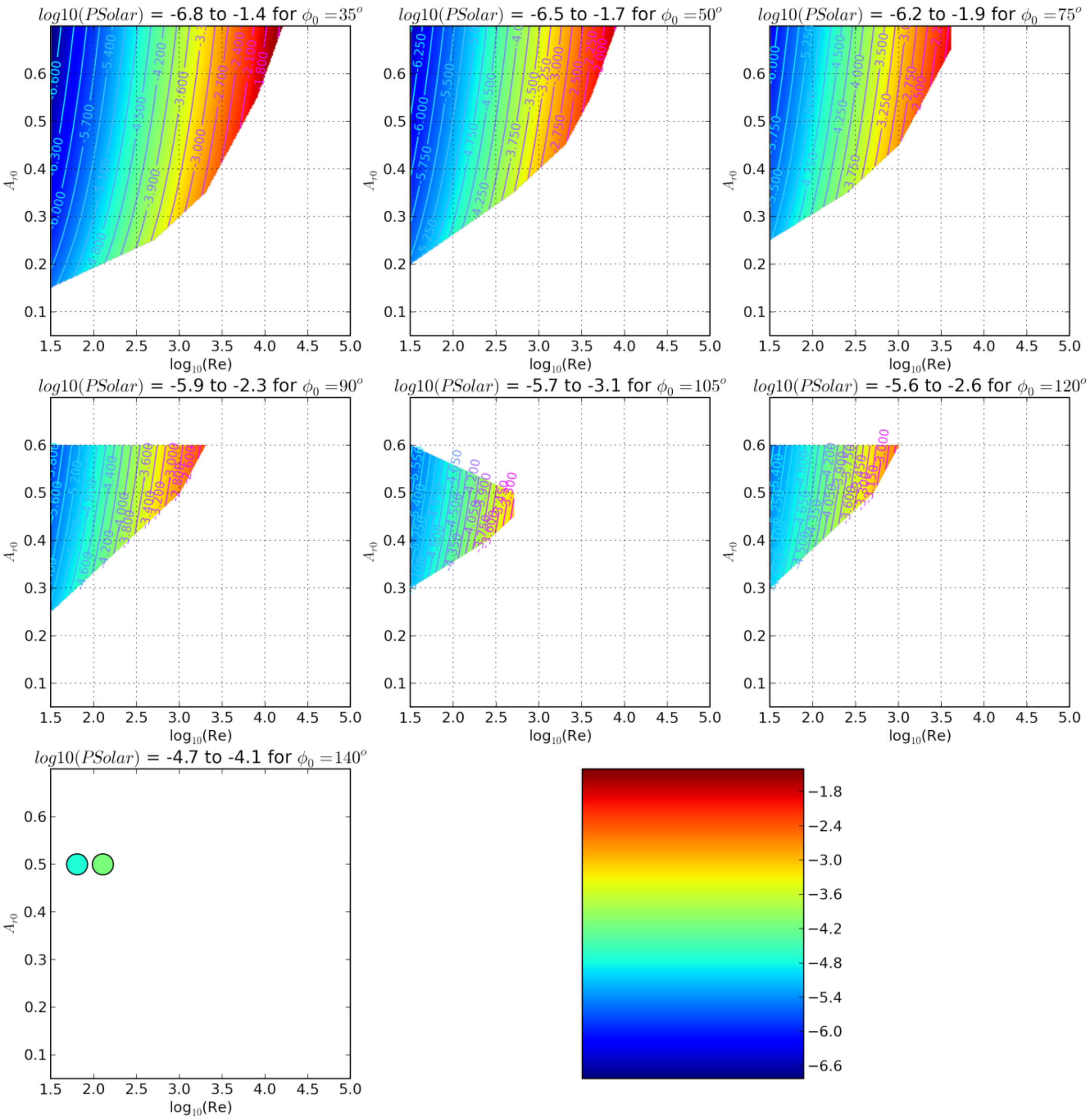


Figure 4-17: $\log_{10}$ of available solar power in W when in 10% of 1 sun

Solar power available scales directly with the square of wing length.

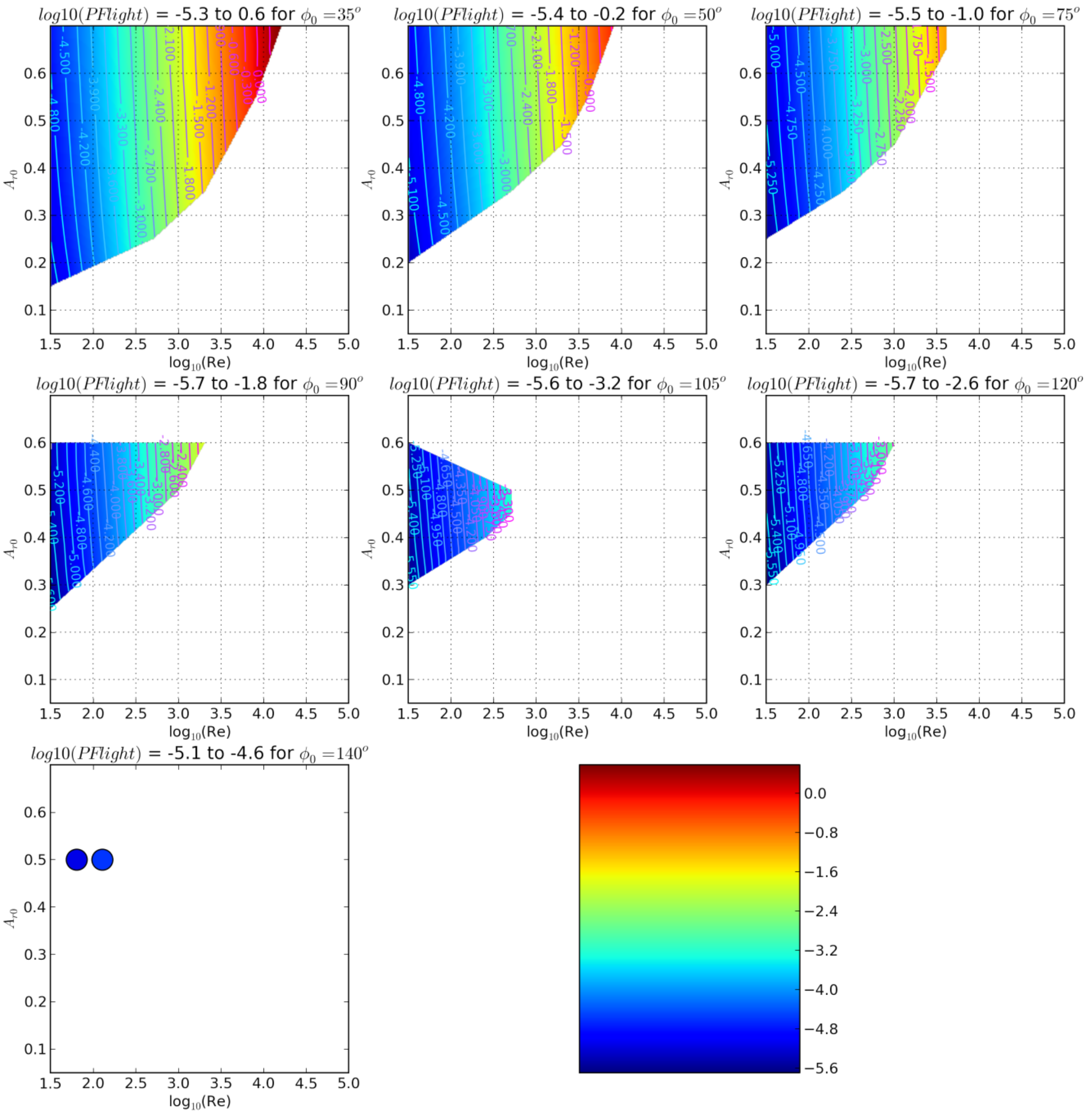


Figure 4-18: $\log_{10}$ of required power for flight, in W, not including electrical losses

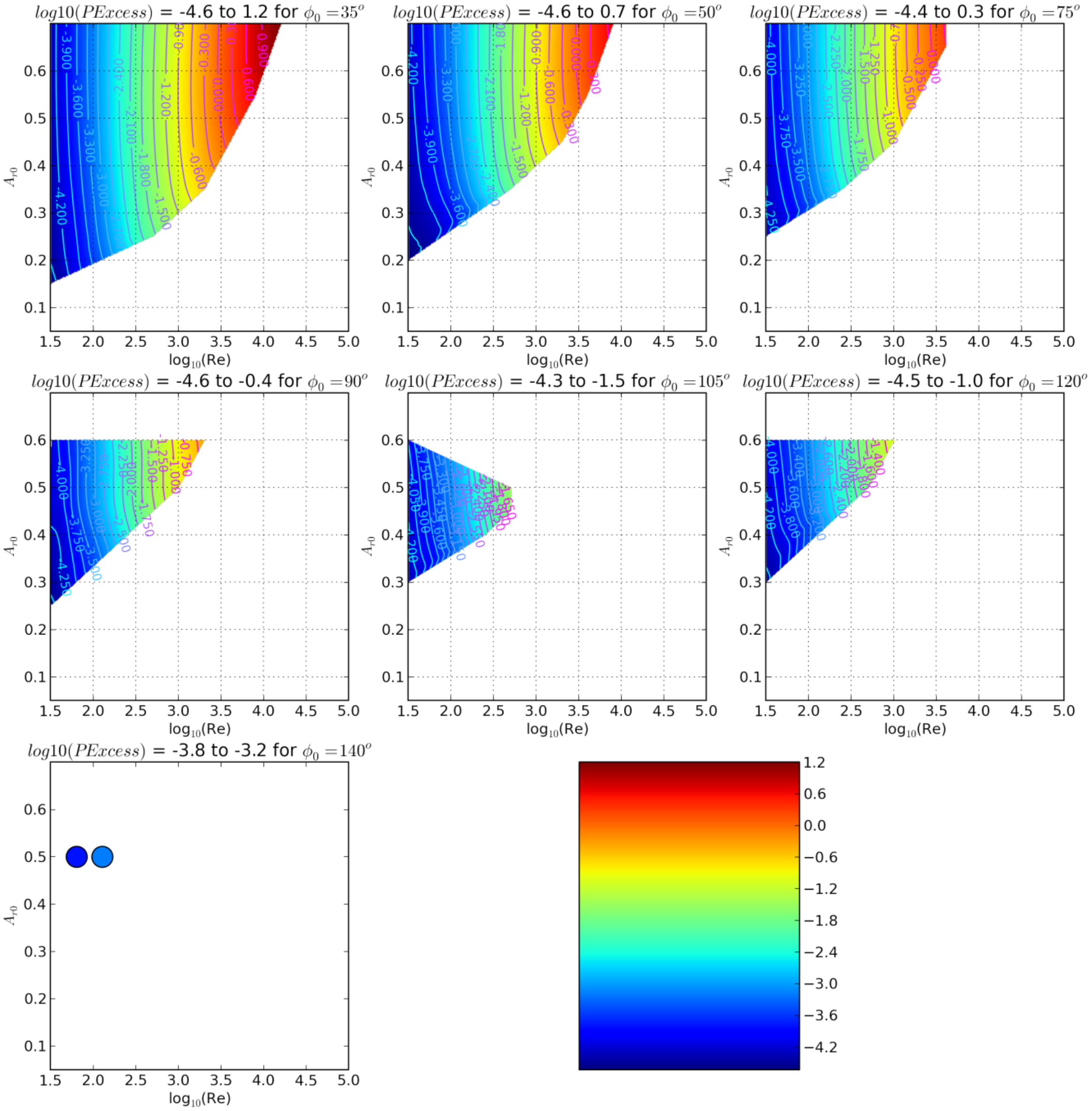


Figure 4-19: $\log_{10}$ of "excess power": the total available power less required power, in W

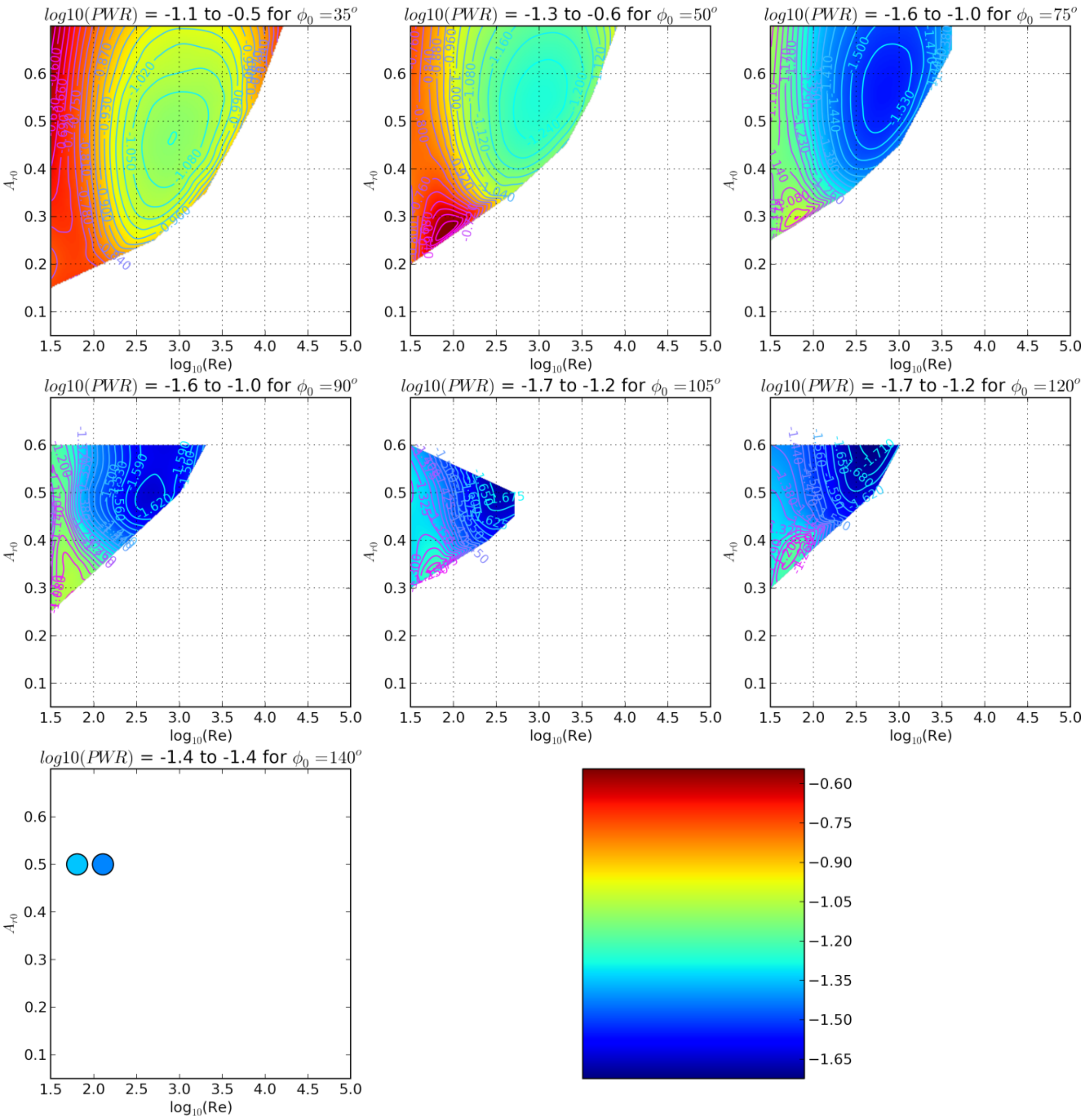


Figure 4-20: log10 of the power ratio: the ratio of required to available power. Like the Wing Weight Ratio, this must be less than 1 for the robot to fly

The lower bending angle designs produce less force for the same Reynolds number and use more power to produce that force. However these designs can be built with shorter wings making them lighter, and for many shapes, able to lift a larger battery than their higher-bending angle counterparts. And so, although the designs with the largest *fraction* of their power not used for flying are have high bending angles, the designs with the largest *magnitude* of power not used for flying have low ones. This persists across different stack input conditions as well, as shown in the following figures for an identical Q:Si structure with double the starting strain, leading to double the γ_{c0}.

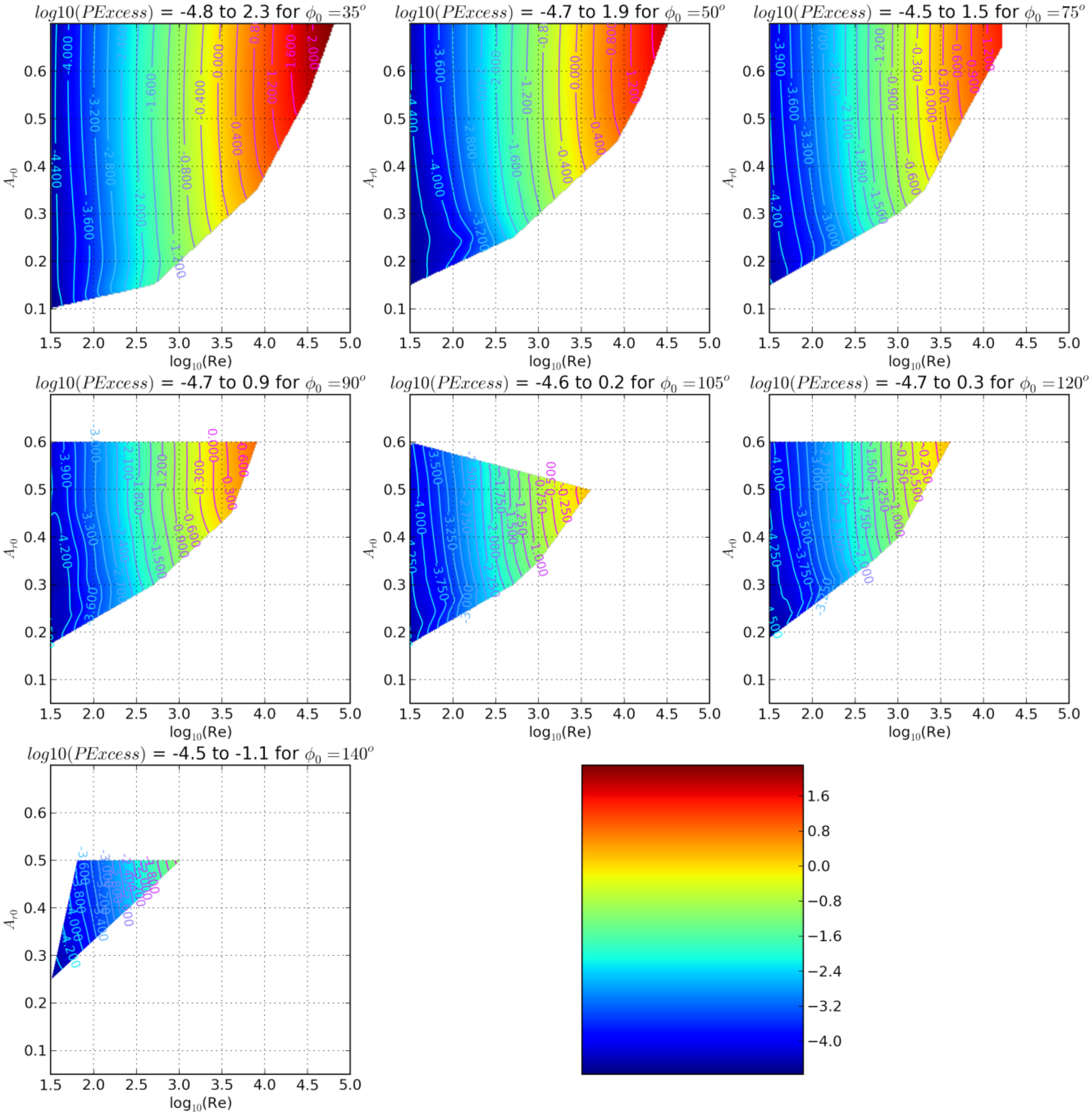


Figure 4-21: Log$_{10}$ of excess power

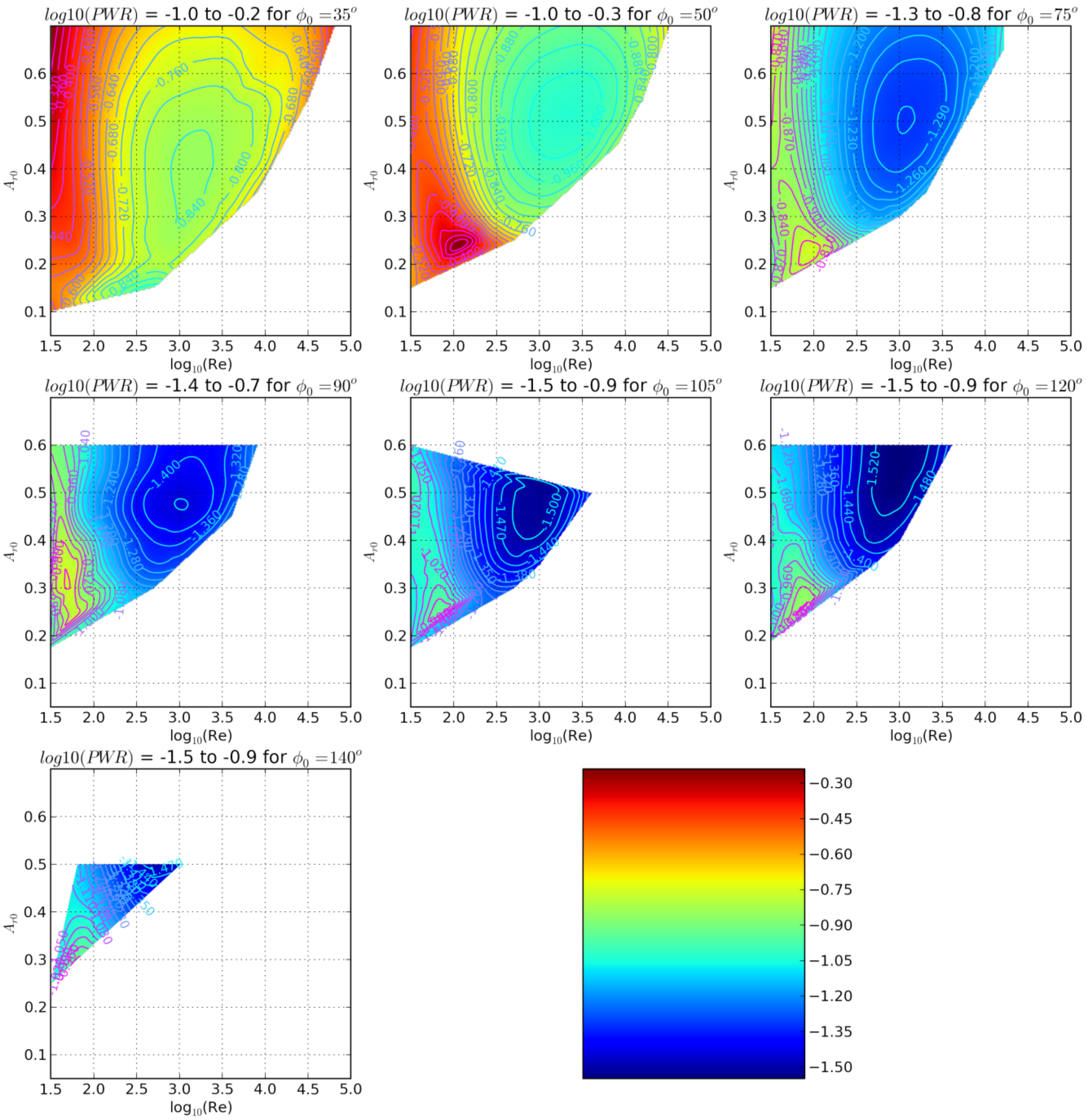


Figure 4-22: Log$_{10}$ of the ratio of dissipated power to available power

For any of the initial bending angles there appears to be a sweet spot where a minimum relative power is used for the wings. This is located near, but not on, the limiting wall for a liftable battery mass, and generally towards the high amplitude, high Reynolds number portion of the plots.

Section 4.4.5 Figures of Merit

Returning to the original built-in curvature of the Q:Si wings, this section determines several figures of merit of the robots which fly.

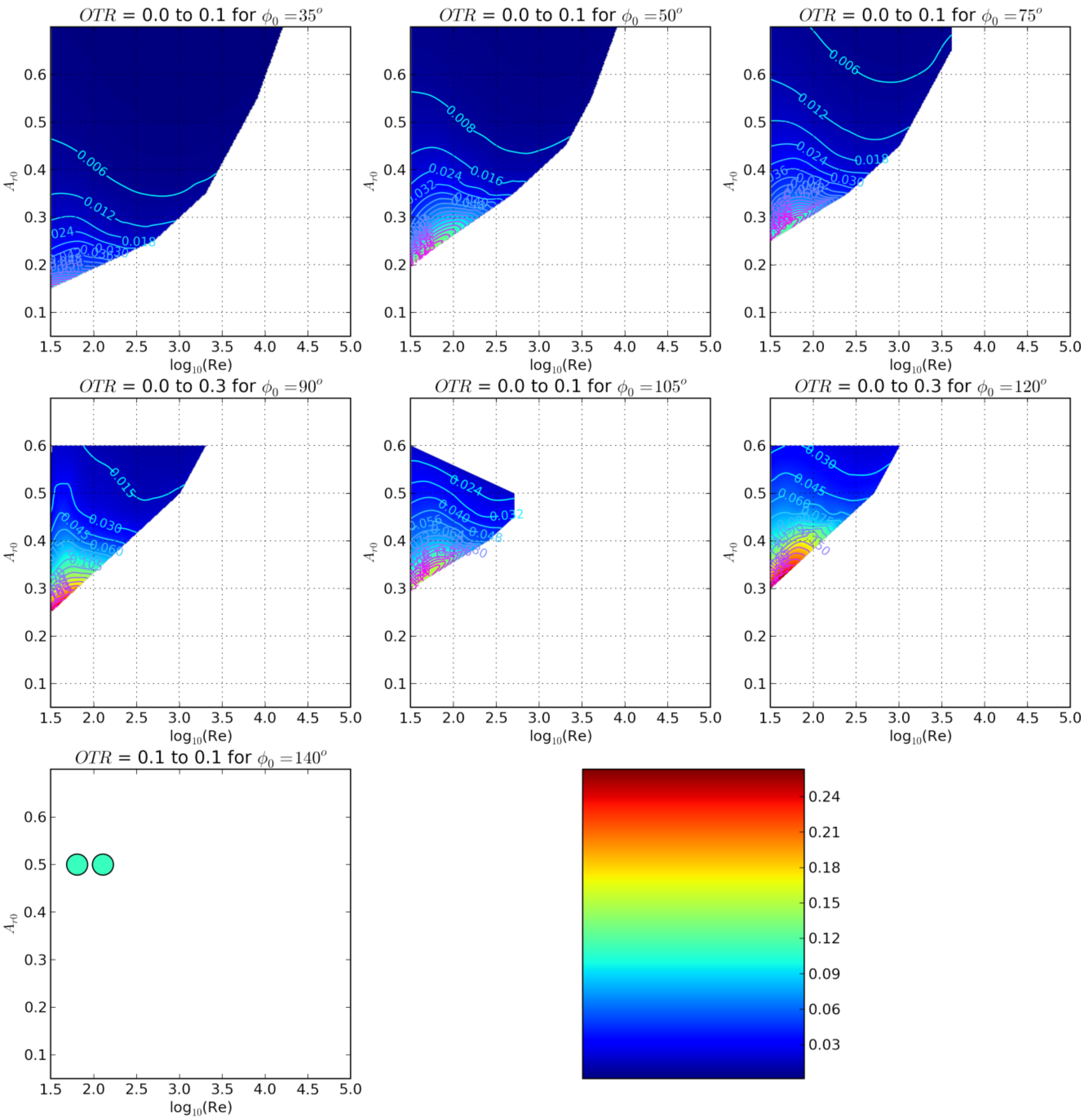


Figure 4-23: Operational time ratio, the fraction of time the robot can spend flying rather than charging when it uses all of its excess power when flying and none of it while charging

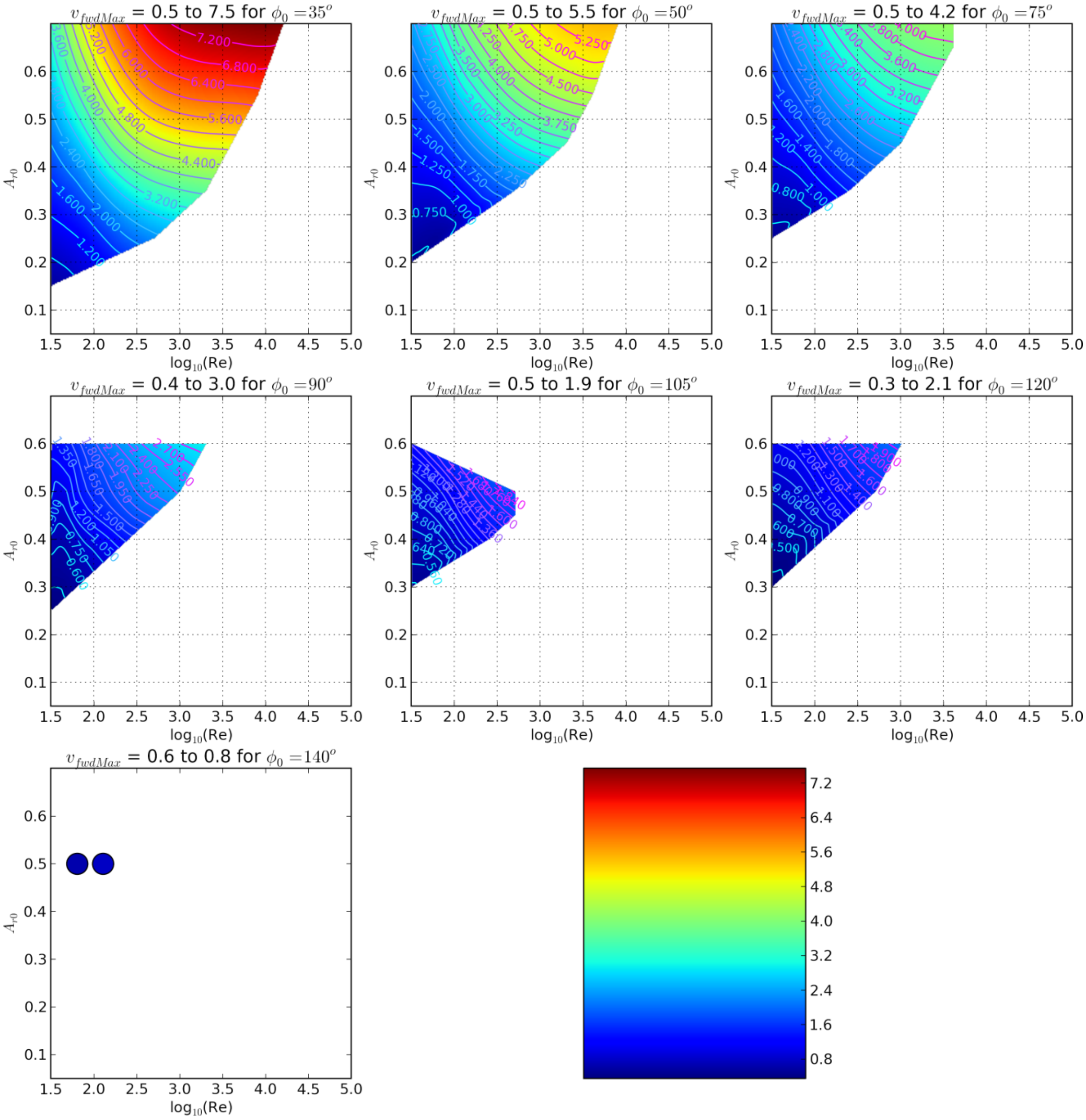


Figure 4-24: Maximum forward velocity, in m/s

The low-curvature designs have nearly zero operational time ratio because their low wing lengths mean their solar power is low. Multiplying the maximum forward velocity and the operational time ratio gives an interesting quantity: the effective velocity over many charges and discharges, while still utilizing all excess power for things besides flying.

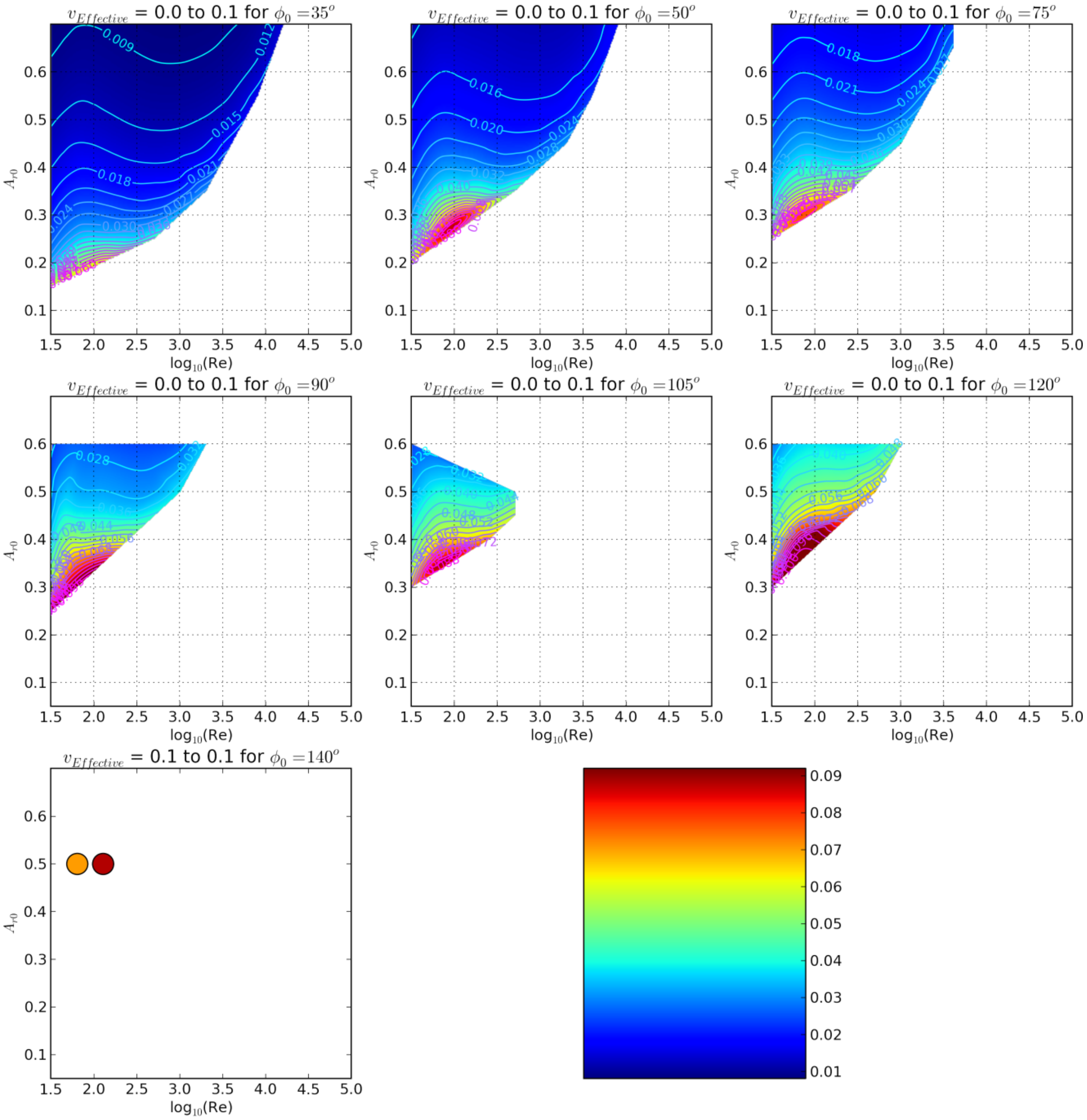


Figure 4-25: Effective forward velocity in m/s. This is the average forward velocity of the robot over many charges and discharges if it uses all of its excess power when flying

Effective velocity favours lower amplitudes and higher initial bending angles. In addition, we can see which robots are best at long-distance travel by plotting the speed a robot has when only drawing energy from its solar power. If solar power alone is sufficient for flight, this is equal to the robot's maximum speed, but if solar power is less than the flight power, it is equal to the this speed multiplied by the fraction of its time the robot can spend flying, which is the ratio of solar power to required power for flight. This "travel velocity" v_{travel} is plotted in the next figure.

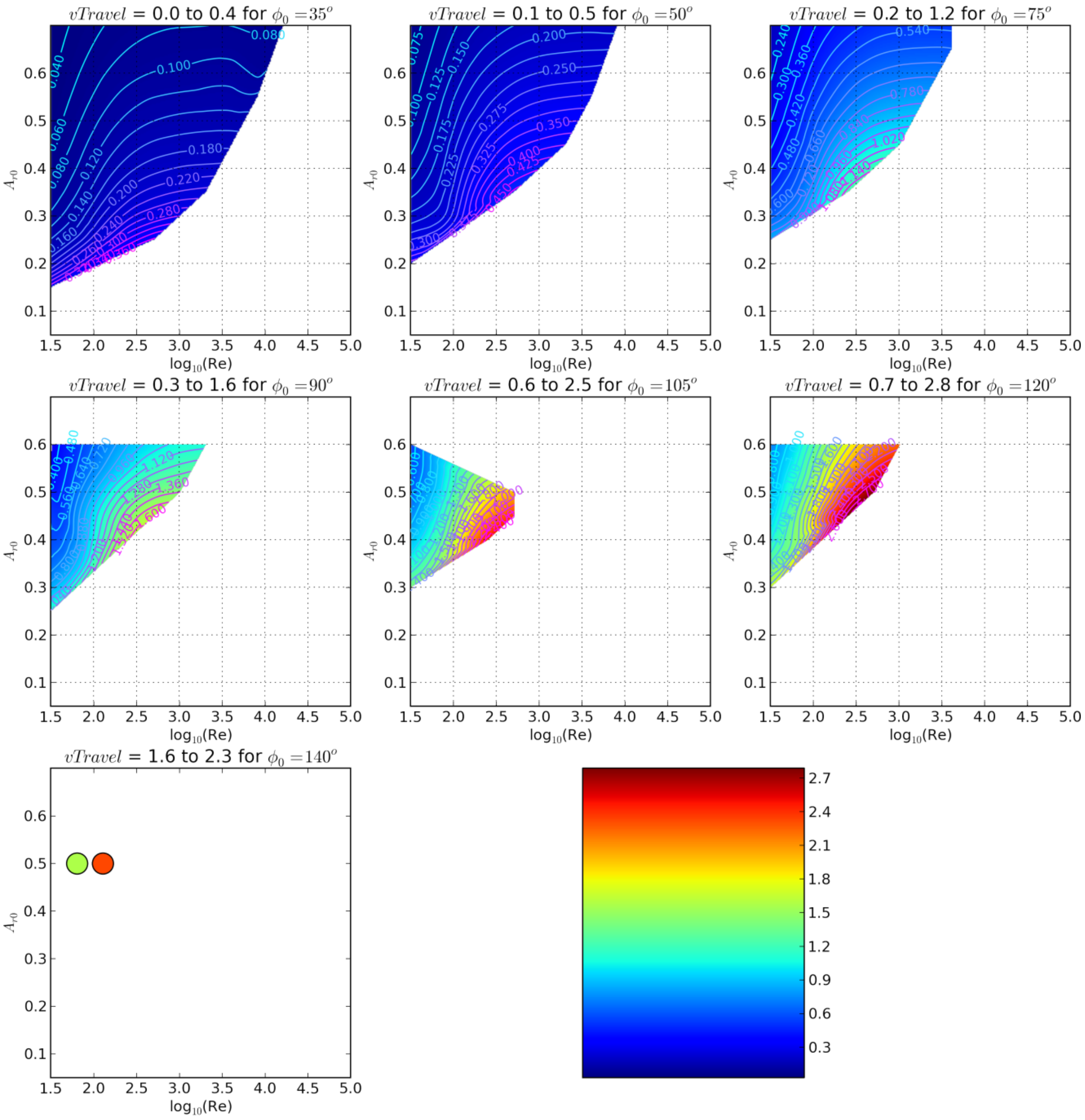


Figure 4-26: Velocity when travelling in m/s is the speed of the robot if it focuses on long-distance travel, spending all of its power to fly and saving excess power rather than using it on other functions

Long-term travel velocity favours higher bending angles and amplitudes, because their larger wings directly increase the solar power, and solar power is the source of the robots' energy.

The VolumeScanAbility is a composite figure of merit that simultaneously accounts for several useful factors of the robot. Assume each robot is doing a scan whose range is 1 m times the square-root of its excess power in W, and that they need to scan a volume of 5 m by 10 m by 10 m, which is 500 m away from where the robots are released. How long would it take a swarm of robots made from a 4" wafer to do this? This time is called the "VolumeScanTime", plotted in Figure 4-27. This is the most complete figure of merit, accounting for a significant travel distance followed by useful operation. Evidently, the best robots from this perspective are the 105^o and 120^o ones, which can accomplish this task in $10^{2.4}$ s = 4 minutes, 11 seconds.

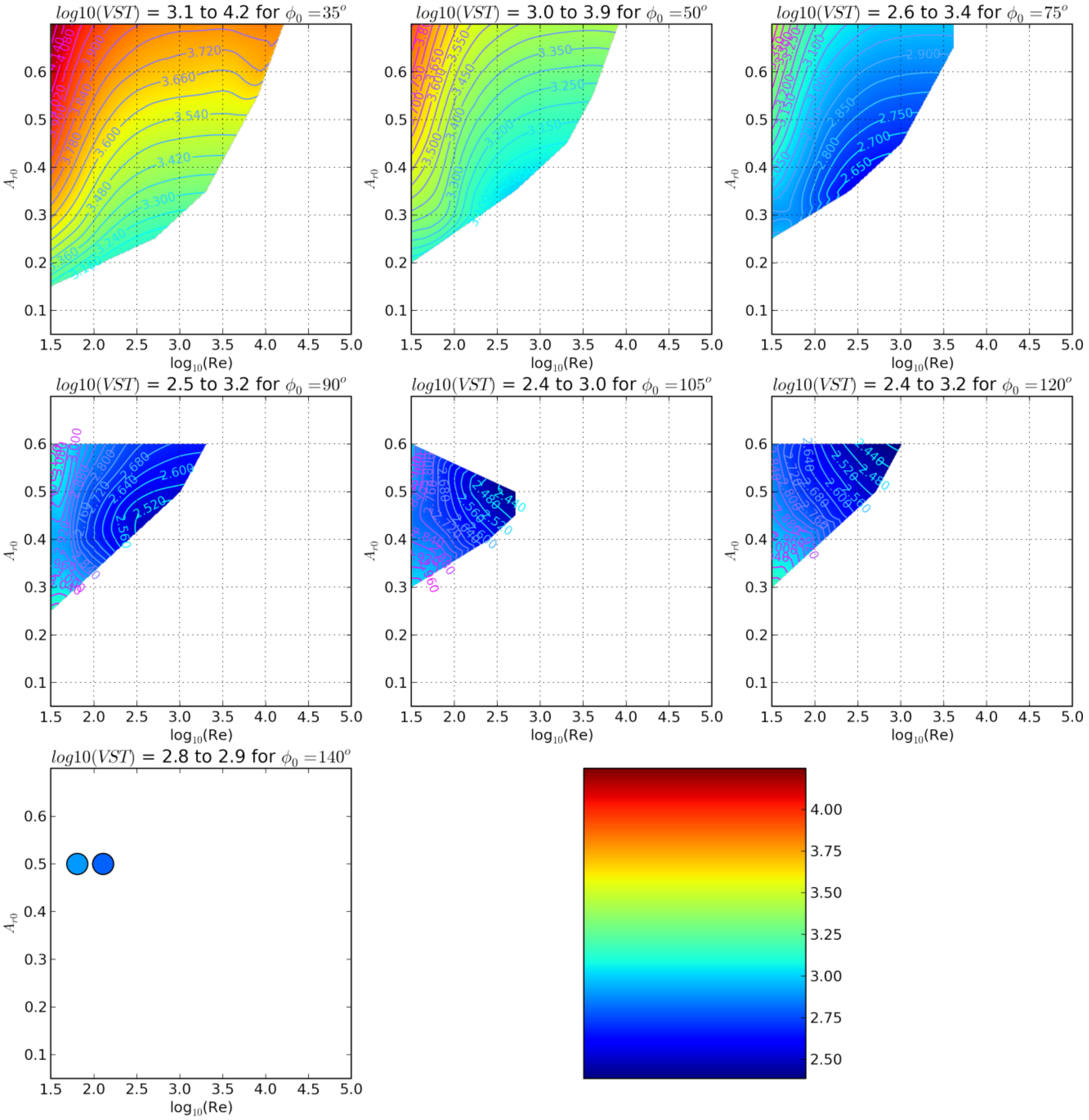


Figure 4-27: $\log_{10}$ of the volume scan time in seconds for a 1-wafer swarm of robots

Section 4.5 Optimization Routine with Inefficiencies

Next, we can attempt to account for overhead by asserting a fixed core dimension, mass, and power requirement. We can get an idea of what is required by using the parameters of the ATtiny4, Atmel's smallest microcontroller, shown in Figure 4-28 [46].

Figure 4-28: Atmel's ATtiny4 microcontroller is only 2 mm by 2 mm by 0.5 mm including the package, and uses only 0.4 mW of power when active at 1 MHz [46]

Based on the packaged dimensions of 2 mm by 2 mm by 0.5 mm, I will assume dimensions of 1 mm by 1 mm and thickness of 100 μm for the core without the package, for a mass of 2e-7 kg. The typical operating current at 2 V in the active mode at 1 MHz is 0.2 mA for a power consumption of 0.4 mW, and 10% of this in the idle mode: 0.04 mW.

These constraints redefine the status of the robots as follows:

1. Battery mass: $m_{batt} = \frac{F_N}{\sqrt{2}g} - m_{wings} - m_{core}$, $m_{core} \equiv 2\text{e-}7$ kg
2. Excess power, flying: $P_{excess} = P_{batt} + P_{solar} - P_{flight} - P_{core}$, $P_{core} \equiv 0.4$ mW
3. Excess power, resting: $P_{resting} = P_{solar} - P_{idlecore}$, $P_{idlecore} = 0.04\text{mW}$

As expected, the constraint on a minimum core mass has had the biggest effect on the smallest robots, eliminating those with Reynolds number lower than 250.

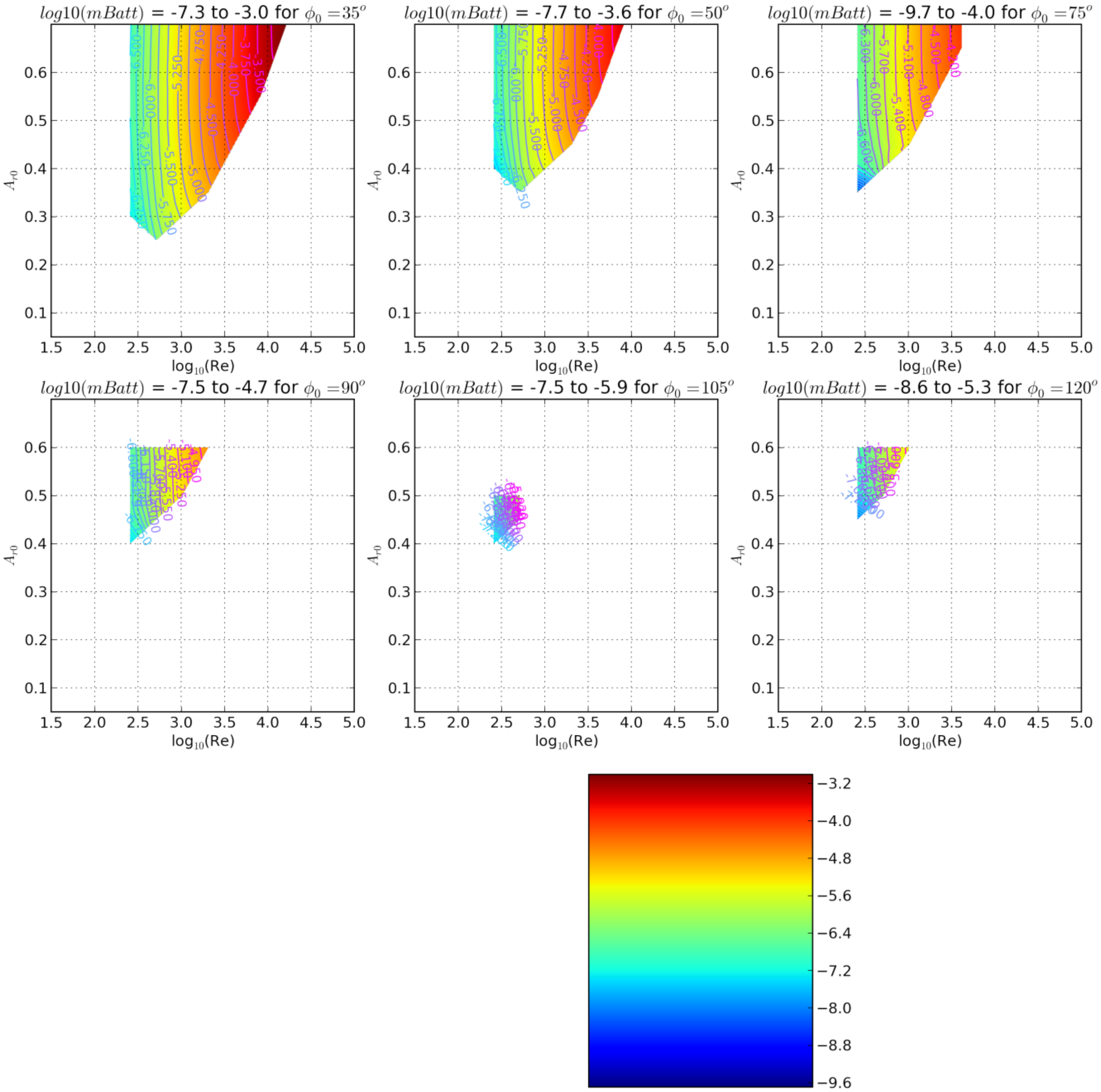


Figure 4-29: $\log_{10}$ of battery mass in kg if lifting a fixed core

Constraining to positive excess power while still power the core further restricts the design, eliminating the highest amplitude simulations at the lowest working Reynolds number for the two smallest bending angles, as shown in Figure 4-30.

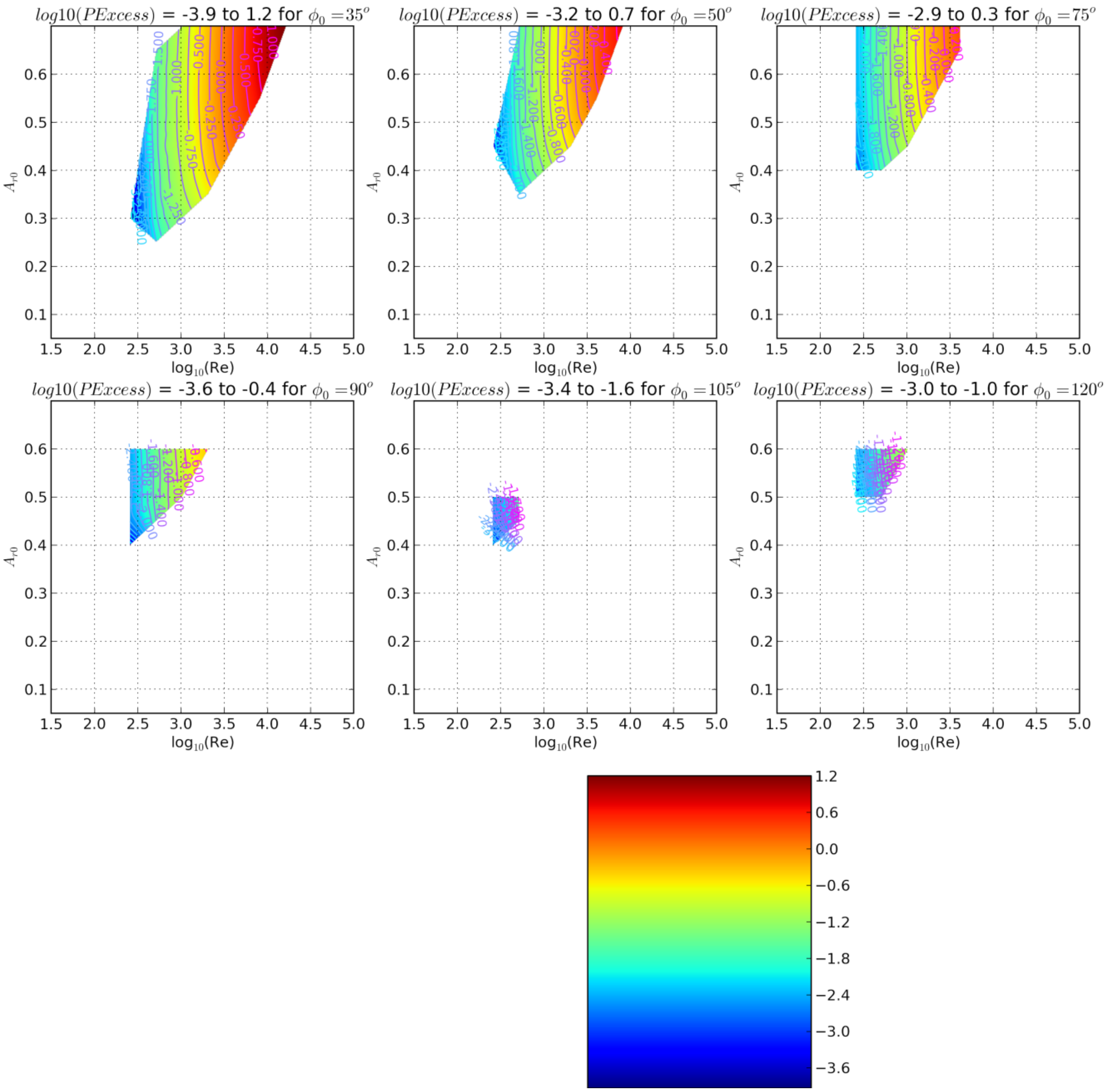


Figure 4-30: $\log_{10}$ of excess power in W for a fixed core

Finally, we can calculate the volume scan time with a fixed core. The fixed core does three things to make the volume scan times worse:

1. it lowers the battery mass by the core mass, lowering the available power,
2. it raises the required power by the core power, and
3. it raises the wafer area required per robot, lowering the number of robots in the swarm produced by a single source wafer.

The volume scan time with a core is plotted in Figure 4-31. The requirement of a fixed core has had little effect on the robot designs with the best volume scan times, leaving them at $10^{2.4}$ seconds.

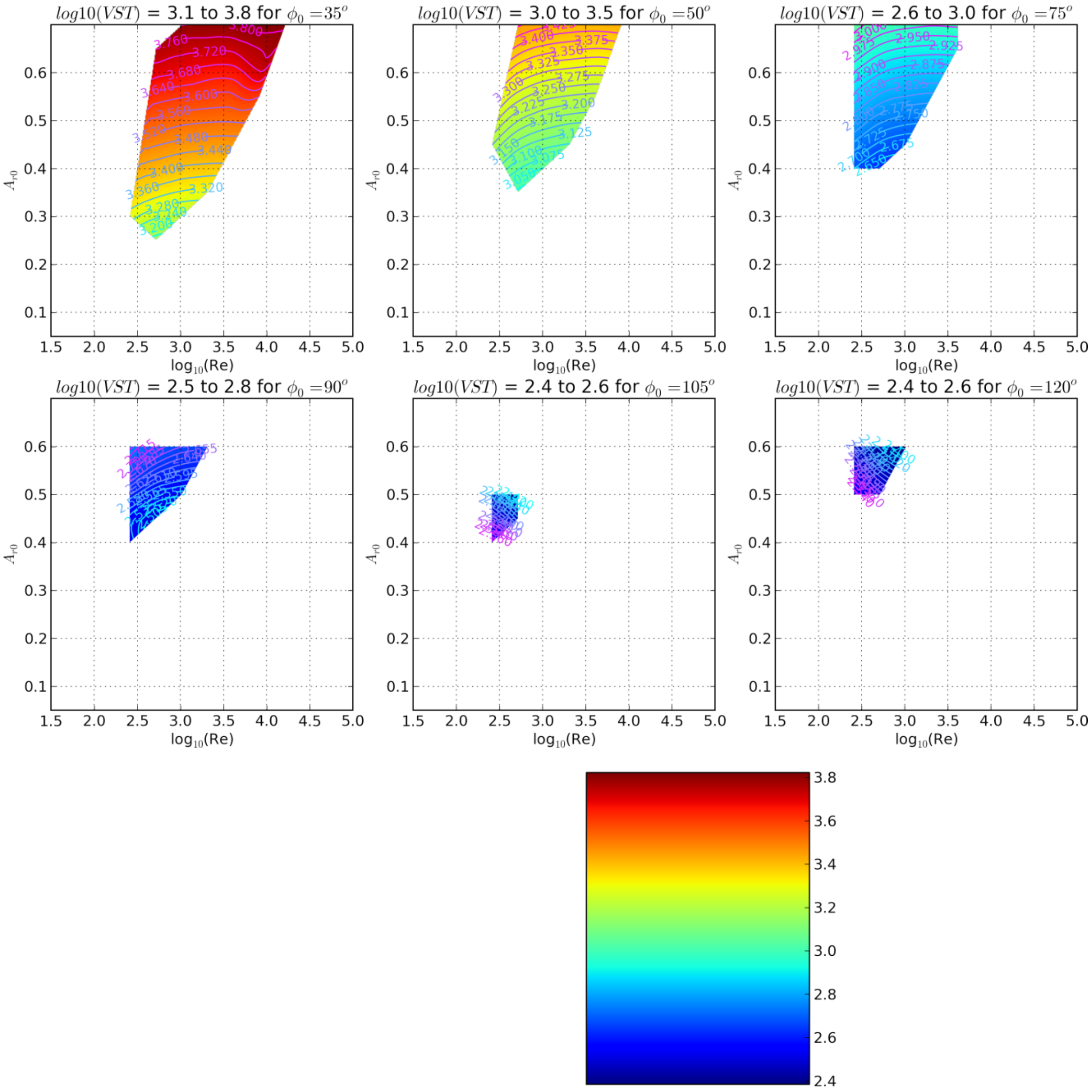


Figure 4-31: $\log_{10}$ of the volume scan time with a fixed core

Of the robots that can lift this core and still fly, what scale of UAV are they? Certainly constraining to lift a 2e-7 kg = 0.2 mg and 1 mm-by-1 mm core puts a lower limit on their UAV class as an XAV, but do any of the predicted robots achieve this range?

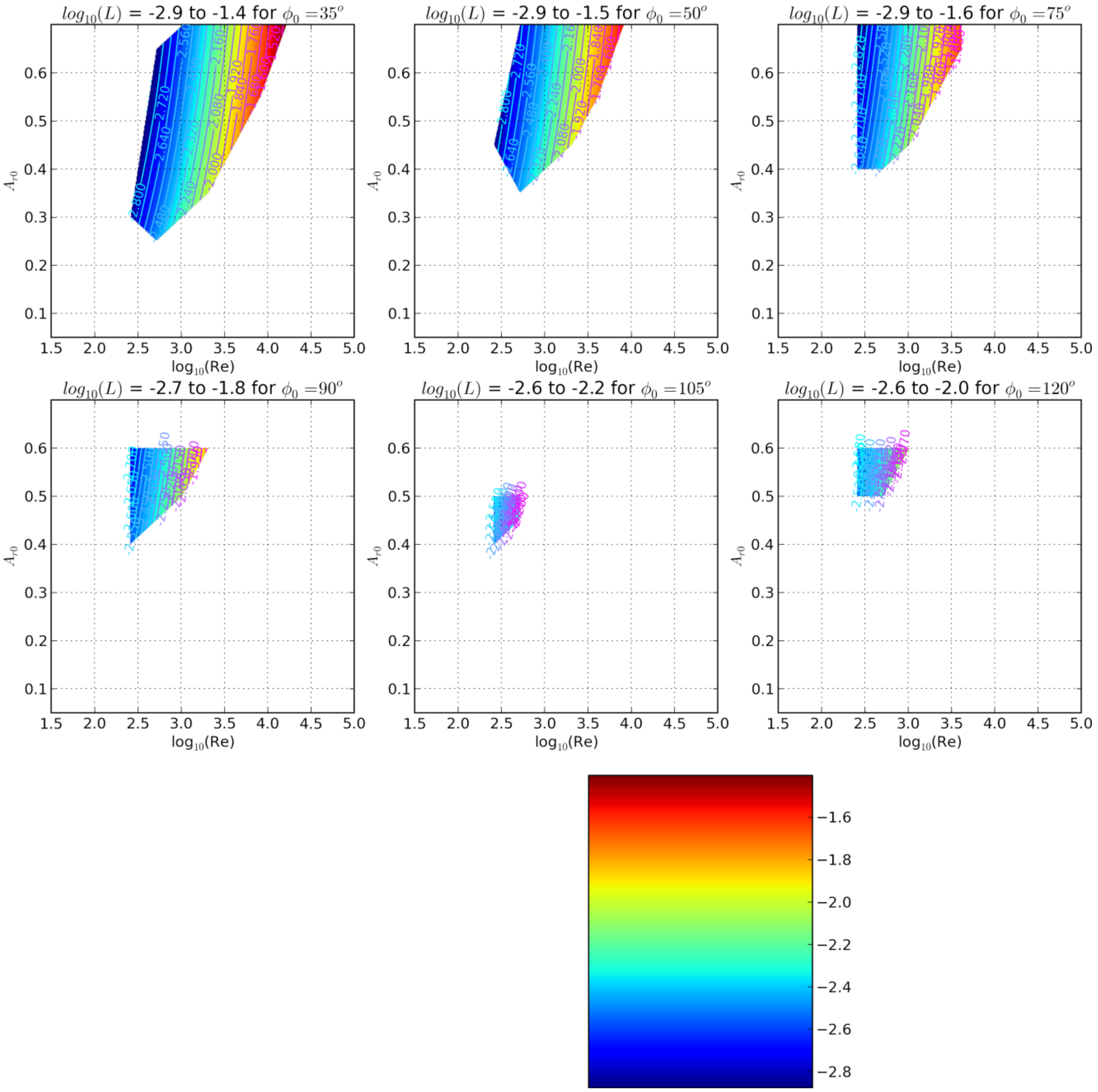


Figure 4-32: $\log_{10}$ of length of working designs with a fixed core

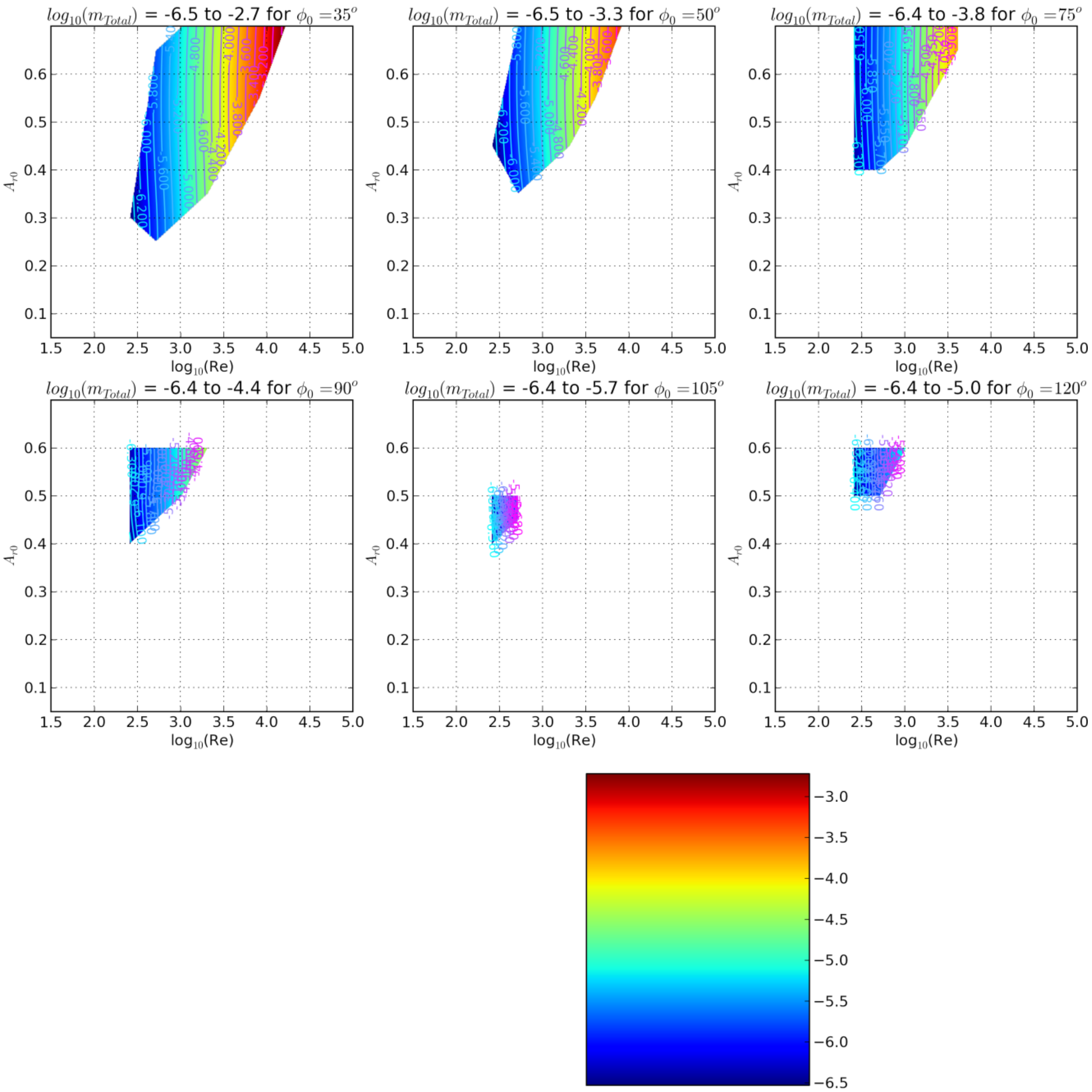


Figure 4-33: $\log_{10}$ of total robot mass of working designs with a fixed core

The best VST designs are mostly affected by amplitude and static bending angle, independent of Reynolds number, while the smallest designs for both length and mass are at the lowest Reynolds number, largely independent of both amplitude and static bending angle. The smallest design which flies corresponds to ϕ_{0g} = 35^o, and has a length of L = 10$^{-2.9}$ m = 1.3 mm and a mass of 10$^{-6.5}$ kg = 0.32 mg, but have a VST of 2290 s. The best VST designs of 251 s are actually not very far behind in line for title of smallest flying robots, with L = 10$^{-2.6}$ kg = 2.5 mm and a mass of 10$^{-6.4}$ kg = 0.40 mg. Referring back to

Table 1-1, these designs are XAVs: Xeno Air Vehicles. If the analytical framework is correct, then this quartz:Si design, which is not necessarily the optimum material choice, can lead flying robots on the scale of a fruit fly even while lifting microcontrollers on the market today.

Section 4.5.1 Discussion

The optimization routine has provided a significant amount of insight that was not anticipated by the drag coefficients in isolation. In Section 3.3 it was easy to dismiss the lower static bending angle wings due to their lower net drag coefficients and higher power drag coefficients but the optimization routine demonstrated that under a variety of design goals the low bending angle simulations are the superior choice. This has all stemmed from applying a fixed built-in curvature constant γ_{c0}, which means the wing's length is directly proportional to bending angle. This constraint has required the larger bending angle simulations to have more massive wings, reducing the battery mass more than their increased force generation increased it. The low bending angle simulations were only unambiguously inferior in one respect: peak voltage and electric field required. For highly dielectric quartz wings, this was not an issue, but there are material choices where this would be more important, pushing the higher curvatures to more desirable situations.

The next section turns to analytically examining the output of the framework in an effort to explore which material parameters are optimal.

Section 4.6 Analytical Scaling Calculations

Is it possible to carry input variables through the calculations and see clearly their effect on the output figures of merit in order to know *a priori* what input parameters we should be designing for? Specifically, which stack structure parameters are optimal?

The thickness, length, and frequency as functions of the simulation and stack structure are:

$$h = \frac{\mu}{\rho_{air}} \frac{4\,\mathrm{Re}}{A_{r0}\lambda^2} \sqrt{\frac{12\rho_m}{E_Y}}$$

$$L = \frac{h\phi_0}{\gamma_{c0}} = \frac{\phi_0}{\gamma_{c0}} \frac{\mu}{\rho_{air}} \frac{4\,\mathrm{Re}}{A_{r0}\lambda^2} \sqrt{\frac{12\rho_m}{E_Y}}$$

$$f = \frac{\lambda^2}{2\pi} \sqrt{\frac{E_Y}{12\rho_m}} f_{resCor} \frac{h}{L^2} = \frac{\lambda^4}{2\pi} \frac{\rho_{air}}{\mu} \frac{E_Y \gamma_{c0}^2}{48\rho_m} \frac{f_{resCor} A_{r0}}{\mathrm{Re}}$$

Thus, for a given CFX simulation, the thickness depends only on $\sqrt{\frac{\rho_m}{E_Y}}$, the square root of the effective wing density over its effective Young's modulus. The length is proportional to the thickness multiplied by $\frac{\phi_0}{\gamma_{c0}}$, i.e., the ratio of bending angle to the curvature constant, $\gamma_{c0} \equiv \frac{\tilde{\mathbf{D}}\tilde{\mathbf{A}}^{-1}}{1-\frac{1}{2}\tilde{\mathbf{D}}\tilde{\mathbf{A}}^{-1}\tilde{\mathbf{B}}}\tilde{\mathbf{C}}_0$, and so for a fixed simulation is proportional to $\frac{1}{\gamma_{c0}}\sqrt{\frac{\rho_m}{E_Y}}$.

The frequency is then proportional to $\frac{E_Y \gamma_{c0}^2}{\rho_m}$.

$$Q = \frac{A_{r0}}{\gamma_{ArDC} \frac{LV}{h^2}}$$

$$Q \equiv \frac{2\pi K_{\max}}{TP_A} = \frac{2 f_{resCor} \rho_m}{\rho_{air} C_{DP} A_{r0}}$$

$$\rightarrow V = \frac{h^2}{L \gamma_{ArDC} Q} A_{r0}$$

By using the second definition of quality factor as an energy ratio, it depends only on the density for a given CFX simulation: higher density leads to higher quality factor.

The voltage required thus depends on:

$$V = \frac{h^2}{L\gamma_{ArDC}\left(\frac{2f_{resCor}\rho_m}{\rho_{air}C_{DP}A_{r0}}\right)}A_{r0}$$

$$= \frac{h^2\rho_{air}C_{DP}A_{r0}^2}{L\gamma_{ArDC}2f_{resCor}\rho_m}$$

Net force:

$F_N = \tfrac{1}{2}\rho_{air}C_{DN} \times v_{rms}^2 A_{wings} = 4\frac{C_{DN}}{2\rho_{air}}\mu^2 \mathrm{Re}^2$ (only a function of the CFX simulation chosen; the 4 is because there are 4 wings total, so

$\mathrm{Re} = \rho_{air}\frac{v_{rms}\sqrt{A_{wing}}}{\mu} \rightarrow v_{rms}^2 A_{wings} = 4v_{rms}^2 A_{wing} = 4\left(\frac{\mathrm{Re}^2\mu^2}{\rho_{air}^2}\right)$)

The battery mass is:

$$m_{batt} = \frac{F_N}{\sqrt{2}g} - m_{wings} - m_{core}$$

where the wing mass is $m_{wings} = 4h\frac{L^2}{4}\rho_m = \rho_m hL^2$, leading to:

$$m_{wings} = \rho_m h\left(\frac{h\phi_0}{\gamma_{c0}}\right)^2$$

$$= \rho_m\frac{\phi_0^2}{\gamma_{c0}^2}\left(\frac{\mu}{\rho_{air}}\frac{4\,\mathrm{Re}}{A_{r0}\lambda^2}\sqrt{\frac{12\rho_m}{E_Y}}\right)^3$$

$$= \phi_0^2\left(\frac{\mu}{\rho_{air}}\frac{4\sqrt{12}\,\mathrm{Re}}{A_{r0}\lambda^2}\right)^3\frac{\rho_m^{2.5}}{\gamma_{c0}^2E_Y^{1.5}}$$

Therefore, all things considered, thanks to the resonant frequency constraint, the wing mass actually scales with density to the power of 2.5 (among other things)!

Ignoring core mass for now, the battery mass is:

$$
\begin{aligned}
m_{batt} &= \frac{F_N}{\sqrt{2}g} - m_{wings} \\
&= \frac{\left(4\dfrac{C_{DN}}{2\rho_{air}}\mu^2 \operatorname{Re}^2\right)}{\sqrt{2}g} - \phi_0^2 \left(\frac{\mu}{\rho_{air}} \frac{4\sqrt{12}\operatorname{Re}}{A_{r0}\lambda^2}\right)^3 \frac{\rho_m^{2.5}}{\gamma_{c0}^2 E_Y^{1.5}} \\
&= \sqrt{2}\frac{C_{DN}}{\rho_{air}g}\mu^2 \operatorname{Re}^2 - \phi_0^2 \left(\frac{\mu}{\rho_{air}} \frac{4\sqrt{12}\operatorname{Re}}{A_{r0}\lambda^2}\right)^3 \frac{\rho_m^{2.5}}{\gamma_{c0}^2 E_Y^{1.5}} \\
&= \operatorname{Re}^2 \left(\sqrt{2}\frac{C_{DN}}{\rho_{air}g}\mu^2 - \operatorname{Re}\phi_0^2 \left(\frac{\mu}{\rho_{air}} \frac{4\sqrt{12}}{A_{r0}\lambda^2}\right)^3 \frac{\rho_m^{2.5}}{\gamma_{c0}^2 E_Y^{1.5}}\right)
\end{aligned}
$$

For the robot to fly, the battery mass must be positive. Since the fluid mechanics simulations reveal that C_{DN} is sub-linear in Re, requiring a positive battery mass results in an upper limit on the Reynolds number for successful robots:

$$
\operatorname{Re}_{\max} = \frac{\sqrt{2}\dfrac{C_{DN}}{\rho_{air}g}\mu^2}{\phi_0^2\left(\dfrac{\mu}{\rho_{air}}\dfrac{4\sqrt{12}}{A_{r0}\lambda^2}\right)^3 \dfrac{\rho_m^{2.5}}{\gamma_{c0}^2 E_Y^{1.5}}} \propto \frac{C_{DN}A_{r0}^3}{\phi_0^2}\frac{\gamma_{c0}^2 E_Y^{1.5}}{\rho_m^{2.5}}
$$

This maximum Re for a positive battery mass explains the A_{r0}-dependent wall observed for the optimization results as log(Re) increases: for a fixed stack, ϕ_0, and A_{r0}, increasing Re increases the wing mass like Re^3 while only increasing the force like Re^2.

For the power balance, we will ignore electrical power losses and the core mass for the time being. The power used is then $P_{required} = P_A$, where

$$
P_A = \tfrac{1}{2}\rho_{air} C_{DP} v_{rms}^3 A_{wings}
$$

and using $v_{rms} = \dfrac{\mu \operatorname{Re}}{\rho_{air}\sqrt{LB}} = 2\dfrac{\mu \operatorname{Re}}{\rho_{air}L}$, this becomes:

$$
\begin{aligned}
P_A &= \tfrac{1}{2}\rho_{air}C_{DP}\left(2\frac{\mu\,\mathrm{Re}}{\rho_{air}L}\right)^3\left(4\frac{L^2}{4}\right)\\
&= 4C_{DP}\frac{\mu^3}{\rho_{air}^2}\frac{\mathrm{Re}^3}{L} = 4C_{DP}\frac{\mu^3}{\rho_{air}^2}\frac{\mathrm{Re}^3}{\left(\frac{\phi_0}{\gamma_{c0}}\frac{\mu}{\rho_{air}}\frac{4\,\mathrm{Re}}{A_{r0}\lambda^2}\sqrt{\frac{12\rho_m}{E_Y}}\right)}\\
&= C_{DP}\frac{\mu^2\,\mathrm{Re}^2\,A_{r0}\lambda^2}{\rho_{air}\phi_0}\frac{\gamma_{c0}}{\sqrt{\frac{12\rho_m}{E_Y}}}
\end{aligned}
$$

The available power is

$$
\begin{aligned}
P_{avail} &= P_{batt} + P_{solar}\\
&= m_{batt}\rho_{pBatt} + A_{wings}\sigma_{pSolar}\\
&= m_{batt}\rho_{pBatt} + L^2\sigma_{pSolar}\\
&= \mathrm{Re}^2\left(\sqrt{2}\frac{C_{DN}}{\rho_{air}g}\mu^2 - \mathrm{Re}\,\phi_0^2\left(\frac{\mu}{\rho_{air}}\frac{4\sqrt{12}}{A_{r0}\lambda^2}\right)^3\frac{\rho_m^{2.5}}{\gamma_{c0}^2E_Y^{1.5}}\right)\rho_{pBatt} + \left(\frac{\phi_0}{\gamma_{c0}}\frac{\mu}{\rho_{air}}\frac{4\,\mathrm{Re}}{A_{r0}\lambda^2}\sqrt{\frac{12\rho_m}{E_Y}}\right)^2\sigma_{pSolar}
\end{aligned}
$$

Meaning the excess power is:

$$
P_{excess} \equiv P_{avail} - P_{required}
$$

$$
= \begin{bmatrix}
\mathrm{Re}^2\left(\sqrt{2}\frac{C_{DN}}{\rho_{air}g}\mu^2 - \mathrm{Re}\,\phi_0^2\left(\frac{\mu}{\rho_{air}}\frac{4\sqrt{12}}{A_{r0}\lambda^2}\right)^3\frac{\rho_m^{2.5}}{\gamma_{c0}^2E_Y^{1.5}}\right)\rho_{pBatt}\\
+\left(\frac{\phi_0}{\gamma_{c0}}\frac{\mu}{\rho_{air}}\frac{4\,\mathrm{Re}}{A_{r0}\lambda^2}\sqrt{\frac{12\rho_m}{E_Y}}\right)^2\sigma_{pSolar}\\
-C_{DP}\frac{\mu^2\,\mathrm{Re}^2\,A_{r0}\lambda^2}{\rho_{air}\phi_0}\frac{\gamma_{c0}}{\sqrt{\frac{12\rho_m}{E_Y}}}
\end{bmatrix}
= \begin{bmatrix}
BatteryPower\\
+SolarPower\\
-FlightPower
\end{bmatrix}
$$

In terms of maximizing output excess power, there are only three unambiguously good parameters:

1. C_{DN}
2. ρ_{pBatt}

3. σ_{pSolar}

And only one unambiguously bad parameter:

1. C_{DP}

In other words, everything else is more complicated! For example, you might imagine that making the wings of a low density material would be better than a heavier material, but is it? Looking at where ρ_m appears here, increasing the wing density:

1. Lowers the battery power (bad)
2. Raises the solar power (good)
3. Lowers the required flight power (good)

Wing stiffness , γ_{EI} , is similarly complicated:

1. Raises the battery power (good)
2. Lowers the solar power (bad)
3. Raises the required flight power (bad)

As is the initial curvature constant, γ_{c0} :

1. Raises the battery power (good)
2. Lowers the solar power (bad)
3. Raises the flight power (bad)

Section 4.6.1 Scaling for the Best Robots

We can gain further insight into these scaling results by examining robots only within the range of steady net and power drag coefficients. As discussed in Section 3.5, the drag coefficients are effectively constant with Re and A_r as long as Re ≥ 1000 and A_{r0} is between 0.45 and 0.65. In fact, in this amplitude and Reynolds number range, as ϕ_0 varies between 75^o and 120^o, C_{DN} is nearly constant at 0.25 while C_{DP} decreases from about 1.6 to 1.3 with increased bending angle and amplitude. As well as being particularly easy due to these constant drag coefficients, this range of Re, A_{r0}, and ϕ_0 is particularly interesting because it is where the some of the best robots are located, as long as the material parameters can make them possible, as presented in Section 4.4. In this section we will analytically examine the analytical framework results in this range of shape parameters. To do so, we will use the three example stack structures from Section 4.2.1.4. Table 4-3 showed the parameters for these stacks.

Once again, thickness, length, and frequency are:

$$h = \frac{\mu}{\rho_{air}} \frac{4\,\mathrm{Re}}{A_{r0}\lambda^2} \sqrt{\frac{12\rho_m}{E_Y}}$$

$$L = \frac{h\phi_0}{\gamma_{c0}} = \frac{\phi_0}{\gamma_{c0}} \frac{\mu}{\rho_{air}} \frac{4\,\mathrm{Re}}{A_{r0}\lambda^2} \sqrt{\frac{12\rho_m}{E_Y}}$$

$$f = \frac{\lambda^2}{2\pi} \sqrt{\frac{E_Y}{12\rho_m}} f_{resCor} \frac{h}{L^2} = \frac{\lambda^4}{2\pi} \frac{\rho_{air}}{\mu} \frac{E_Y \gamma_{c0}^2}{48\rho_m} \frac{f_{resCor} A_{r0}}{\mathrm{Re}}$$

The restriction of Re ≥ 1000 and $A_r \leq 0.65$ places a stack-dependent lower limit on wing thickness of

$$h \geq \frac{\mu}{\rho_{air}} \frac{4\left(\mathrm{Re}_{\max}\right)}{\left(A_{r0\min}\right)\lambda^2} \sqrt{\frac{12\rho_m}{E_Y}}$$

For the quartz:Si, Quartz:SU-8, and InGaP stacks of Table 4-3, this lower thickness limit works out to 0.16, 0.36, and 0.19 μm, respectively.

Battery mass is

$$m_{batt} = \frac{F_N}{\sqrt{2}g} - m_{wings}$$

$$= \mathrm{Re}^2 \left(\sqrt{2} \frac{C_{DN}}{\rho_{air} g} \mu^2 - \mathrm{Re}\,\phi_0^2 \left(\frac{\mu}{\rho_{air}} \frac{4\sqrt{12}}{A_{r0}\lambda^2} \right)^3 \frac{\rho_m^{2.5}}{\gamma_{c0}^2 E_Y^{1.5}} \right)$$

Substituting known values of the constants, including $C_{DN} = 0.25$, this equation becomes (in SI units):

$$m_{batt} = \mathrm{Re}^2\left(1.052\text{e-}11 - 2.367\text{e-}13 \times \mathrm{Re}\frac{\phi_0^2}{A_{r0}^3}\frac{\rho_m^{2.5}}{\gamma_{c0}^2 E_Y^{1.5}}\right)$$

The battery mass depends on the stack parameter ratio $\frac{\rho_m^{2.5}}{\gamma_{c0}^2 E_Y^{1.5}}$, which for the three stacks is 811e-6, 607e-5, and 310e-5 [in SI units: $\frac{\left(\mathrm{kg/m^3}\right)^{2.5}}{(1)^2\left(\mathrm{Pa}\right)^{1.5}} = \frac{\mathrm{kg^{0.5}s^4}}{\mathrm{m^{5.5}}}$]. This difference in stack parameter ratios means that the Q:Si wings are four times lighter than the InGaP wings, which themselves are twice as light as the Q:SU8 wings for any choice of shape parameters. Qualitatively, the Q:Si wings do much better than the InGaP wings because they are nearly half the density but have a similar curvature constant and stiffness (and the wing mass scales with wing density to the power of 2.5 due to the resonant frequency constraint). Even though the Q:SU8 wings have a lower density and higher curvature than the Q:Si wings, the SU-8 has made their stiffness so much lower that they need to be significantly thicker than the other stacks to have the same shape parameters.

Requiring positive battery mass sets an upper limit on the Reynolds number, but there is actually a subtler limit before this: if the wings are much more massive than the rest of the robot (i.e., the battery) then the wings would move the fuselage up and down an appreciable amount as they flap, making their movement in the air significantly different than what the simulations modelled. Examining the battery mass equation reveals that the ratio of wing mass to total mass is proportional to Re. By differentiating the battery mass equation with respect to Reynolds number, we can see that for a fixed stack and ϕ_0^2 / A_{r0}^3, the battery mass reaches a maximum value at $\mathrm{Re}_{m_{batt}Max} = \frac{2 \times 1.052\text{e-}11}{3 \times 2.367\text{e-}13\frac{\phi_0^2}{A_{r0}^3}\frac{\rho_m^{2.5}}{\gamma_{c0}^2 E_Y^{1.5}}}$, when the wings account for 2/3 of the total robot mass, meaning they are twice as massive as the battery. For the wings to be reasonably steady against the air, we need the battery to at least match the wing mass, which gives a more restrictive maximum Reynolds number of:

$$\mathrm{Re}_{\max} = \frac{1.052\text{e-}11}{2 \times 2.367\text{e-}13 \dfrac{\phi_0^2}{A_{r0}^3} \dfrac{\rho_m^{2.5}}{\gamma_{c0}^2 E_Y^{1.5}}} \equiv \frac{22.2}{\eta_{shape}\eta_{stack}}$$

where we have broken the formula into one parameter that depends on the stack, $\eta_{stack} \equiv \dfrac{\rho_m^{2.5}}{\gamma_{c0}^2 E_Y^{1.5}}$, and one that depends on the shape, $\eta_{shape} \equiv \dfrac{\phi_0^2}{A_{r0}^3}$.

This maximum Reynolds number increases with the cube of amplitude and decreases with the square of bending angle. Table 4-6 shows $\log_{10}$ of this maximum Re vs. ϕ_0 for each stack structure for the maximum optimum amplitude of 0.65, and illustrates that the battery mass-determining stack parameter ratio, η_{stack}, has a large effect on whether that stack will lead to wings that are light enough for each bending angle.

Table 4-6: Re$_{\max}$ for $m_{batt} \leq m_{wings}$ with A_{r0} = 0.65

Stack\\ϕ_0	η_{stack}	75°	90°	105°	120°
η_{shape}:		4.06	5.84	7.95	10.38
Q:Si	8.11e-4	6759	4694	3448	2640
Q:SU8	6.07e-3	902	627	460	352
InGaP	3.11e-3	1769	1228	902	691

Looking at this another way, if we solve $\mathrm{Re}_{\max} = \dfrac{22.2}{\eta_{shape}\eta_{stack}}$ for the battery mass-determining stack parameter ratio, we can calculate the maximum value of $\eta_{stack} \equiv \dfrac{\rho_m^{2.5}}{\gamma_{c0}^2 E_Y^{1.5}}$ for each bending angle to lead to robot designs with Re ≥ 1000 and $A_{r0} \leq 0.65$. Table 4-7 shows the result, and agrees with the results for Table 4-6 given the ratios for the example stacks, 811e-4, 607e-3, and 310e-3.

Table 4-7: Maximum value of the mass-determining stack parameter ratio η_{stack} (in SI units) to allow Re ≥ 1000

ϕ_0	75°	90°	105°	120°
$A_r \leq 0.65$	5.48E-03	3.81E-03	2.80E-03	2.14E-03
$A_r \leq 0.45$	2.63E-03	1.82E-03	1.34E-03	1.03E-03

So, the battery mass scaling results give us bending angle-dependent upper limit on $\eta_{stack} \equiv \dfrac{\rho_m^{2.5}}{\gamma_{c0}^2 E_Y^{1.5}}$ for working in the best drag-coefficient regions while keeping the battery mass at least 50% of the wing mass. And for each simulation, a lower value of this ratio leads to a larger battery mass. Still, this doesn't necessarily mean that stacks should strive to minimize this ratio. To learn more, we need to examine the power balance constraint.

The excess power [calculated in Section 4.6] is given by

$$P_{excess} \equiv P_{avail} - P_{required}$$

$$= \begin{bmatrix} \mathrm{Re}^2 \left(\sqrt{2} \dfrac{C_{DN}}{\rho_{air} g} \mu^2 - \mathrm{Re}\, \phi_0^2 \left(\dfrac{\mu}{\rho_{air}} \dfrac{4\sqrt{12}}{A_{r0}\lambda^2} \right)^3 \dfrac{\rho_m^{2.5}}{\gamma_{c0}^2 E_Y^{1.5}} \right) \rho_{pBatt} \\ + \left(\dfrac{\phi_0}{\gamma_{c0}} \dfrac{\mu}{\rho_{air}} \dfrac{4\,\mathrm{Re}}{A_{r0}\lambda^2} \sqrt{\dfrac{12\rho_m}{E_Y}} \right)^2 \sigma_{pSolar} \\ -C_{DP} \dfrac{\mu^2\, \mathrm{Re}^2\, A_{r0}\lambda^2}{\rho_{air}\phi_0} \dfrac{\gamma_{c0}}{\sqrt{\dfrac{12\rho_m}{E_Y}}} \end{bmatrix} = \begin{bmatrix} BatteryPower \\ +SolarPower \\ -FlightPower \end{bmatrix}$$

The robot needs to have a positive excess power to be able to fly. This equation reveals that within the sufficient Re and A_{r0} ranges that the drag coefficients are constant, the excess power is an increasing function of Re as long as the battery mass is. Therefore, the maximum excess power for a fixed stack, A_{r0}, and ϕ_0 occurs at the maximum Re from the battery mass balance: $\mathrm{Re}_{\max} = \dfrac{22.2}{\eta_{shape}\eta_{stack}}$ (the max battery mass itself is $m_{battMax} = \left(5.261\text{e-}12\ \mathrm{kg}\right)\mathrm{Re}_{\max}^2$). Substituting this Reynolds number for the largest allowable battery, the largest possible excess power is:

$$P_{excessMax} = \left[\begin{array}{l} \left(\dfrac{\sqrt{2}C_{DN}\mu^2}{2\rho_{air}g}\right)\mathrm{Re}_{\max}^2\ \rho_{pBatt} \\ +\left(\dfrac{\phi_0}{\gamma_{c0}}\dfrac{\mu}{\rho_{air}}\dfrac{4\,\mathrm{Re}_{\max}}{A_{r0}\lambda^2}\sqrt{\dfrac{12\rho_m}{E_Y}}\right)^2 \sigma_{pSolar} \\ -C_{DP}\,\dfrac{\mu^2\,\mathrm{Re}_{\max}^2\,A_{r0}\lambda^2}{\rho_{air}\phi_0}\dfrac{\gamma_{c0}}{\sqrt{\dfrac{12\rho_m}{E_Y}}} \end{array}\right]$$

$$= \left[\begin{array}{l} 5.198\text{e-}5\,\dfrac{A_{r0}^2}{\phi_0^4}\dfrac{\gamma_{c0}^4 E_Y^3}{\rho_m^5} \\ +4.726\text{e-}5\,\dfrac{1}{\phi_0^2}\dfrac{\gamma_{c0}^2 E_Y^2}{\rho_m^4} \\ -1.464\text{e-}7C_{DP}\,\dfrac{A_{r0}^3}{\phi_0^5}\dfrac{\gamma_{c0}^5 E_Y^{3.5}}{\rho_m^{5.5}} \end{array}\right] = \left[\begin{array}{l} 1.052\text{e-}7 \\ +9.566\text{e-}8\,\dfrac{\phi_0^2}{A_{r0}^2}\dfrac{\rho_m}{\gamma_{c0}^2 E_Y} \\ -2.963\text{e-}10C_{DP}\,\dfrac{A_{r0}}{\phi_0}\dfrac{\gamma_{c0}E_Y^{0.5}}{\rho_m^{0.5}} \end{array}\right]\mathrm{Re}_{\max}^2$$

Besides the variation in C_{DP} with ϕ_0, the maximum excess power turns out to depend only on maximum Reynolds number and the parameter $\xi \equiv \dfrac{A_{r0}}{\phi_0}\dfrac{\gamma_{c0}E_Y^{0.5}}{\rho_m^{0.5}}$. For a fixed maximum Re, the ratio of required flight power to battery power increases linearly with ξ while the ratio of available solar power to battery power decreases quadratically with it. For both of these reasons, minimizing ξ for fixed $\mathrm{Re}_{\max}$ is beneficial. This is nontrivial because $\mathrm{Re}_{\max} \propto \dfrac{A_{r0}^3}{\phi_0^2}\dfrac{\gamma_{c0}^2 E_Y^{1.5}}{\rho_m^{2.5}}$ so every parameter that increases $\mathrm{Re}_{\max}$ also increases ξ. Nevertheless, we can say that for a fixed stack, $\mathrm{Re}_{\max}$ is constant for fixed $\dfrac{A_{r0}^3}{\phi_0^2}$, in which case $\phi_0 \propto \dfrac{A_{r0}^{1.5}}{\mathrm{Re}_{\max}^{0.5}}$, meaning $\xi \propto \dfrac{A_{r0}}{A_{r0}^{1.5}} \propto \dfrac{1}{\sqrt{A_{r0}}}$ and therefore we want to *maximize* A_{r0} in order to minimize ξ for a fixed $\mathrm{Re}_{\max}$. Therefore, for any fixed stack and from any starting point for ϕ_0 and A_{r0} we can increase the excess power, reduce the flight power, and reduce the charge time while holding the battery mass constant if we follow the $\phi_0 \propto A_{r0}^{1.5}$ curve to its maximum value in the efficient parameter space of $0.45 \le A_{r0} \le 0.65$ and $75° \le \phi_0 \le 120°$. Depending on the desired Reynolds number, this means the best simulations should either end up with $A_{r0} = 0.65$ or $\phi_0 = 120°$.

Positive excess power requires $1.052\text{e-}7 + 9.566\text{e-}8\xi^{-2} - 2.963\text{e-}10 C_{DP}\xi > 0$, which places a maximum value on ξ. Over the efficient parameter space the shape parameter's contribution to ξ ranges from $\xi_{Shape} \equiv \dfrac{A_{r0}}{\phi_0} = 0.215$ at $\phi_0 = 120°, A_{r0} = 0.45$ to $\xi_{Shape} = 0.497$ at $\phi_0 = 75°, A_{r0} = 0.65$. Referring to Figure 3-13, $C_{DP} = 1.6$ at both extrema.

For the three example stacks, $\xi_{Stack} \equiv \dfrac{\gamma_{c0} E_Y^{0.5}}{\rho_m^{0.5}}$ = 22.4, 11.4, and 16.3, respectively. These enable computing the extrema on the maximum powers per Re^2 at $\mathrm{Re}_{\max}$ to see how serious the excess power restriction is. The results are in Table 4-8: the excess power per Re is not a very sensitive function of ξ. Over the entire range of input parameters, P_{Excess} only ranges from 1.01e-7/Re^2 to 1.20e-7/Re^2. Thus, the P_{Excess} constraint is not an issue next to the flight constraint over the efficient parameter space.

Table 4-8: Range of maximum output powers per Re^2 for ξ in the range of the example stacks and efficient shapes

	ξ_{stack}	ξ_{shape}	ξ	P_{Batt}/ Re^2	P_{Solar}/ Re^2	P_{Flight}/ Re^2	P_{Excess}/ Re^2
Q:Si	22.4	0.215	4.80	1.05E-07	4.14E-09	-2.28E-09	1.07E-07
Q:SU8	11.4	0.215	2.44	1.05E-07	1.60E-08	-1.16E-09	1.20E-07
InGaP	16.3	0.215	3.50	1.05E-07	7.80E-09	-1.66E-09	1.11E-07
Q:Si	22.4	0.497	11.10	1.05E-07	7.76E-10	-5.26E-09	1.01E-07
Q:SU8	11.4	0.497	5.65	1.05E-07	3.00E-09	-2.68E-09	1.06E-07
InGaP	16.3	0.497	8.09	1.05E-07	1.46E-09	-3.84E-09	1.03E-07

In addition, Table 4-8 lists the effect on the solar power, which scales with ξ^{-2}. Although the range of ξ does not have a very large effect on the excess power at the maximum battery size, it does effect the time it takes to charge that battery. Smaller values of ξ lead to larger wings at the max battery Re, more available solar power, and shorter charge times. The battery power used in Table 4-8 is for a 16.25 s discharge time, meaning charge times between 107 s for ξ = 2.44 and 2200 s for ξ = 11.1. The solar power mentioned is for 25% efficiency in 100 W/m^2 lighting. These long charge times illustrate the real importance of reducing ξ: reducing robot down-time.

To summarize, the optimum robot in terms of excess power occurs for a maximized battery mass, which occurs at a maximized value of $\mathrm{Re}_{\max} \propto \eta^{-1} \equiv \frac{A_{r0}^3}{\phi_0^2}\frac{\gamma_{c0}^2 E_Y^{1.5}}{\rho_m^{2.5}}$. At this value, the excess power is proportional to $\mathrm{Re}_{\max}^2$, and is further increased by minimizing $\xi \equiv \frac{A_{r0}}{\phi_0}\frac{\gamma_{c0} E_Y^{0.5}}{\rho_m^{0.5}}$, resulting in reduced required flight power and increased available solar power. For the stack examples and shape parameter ranges considered, Re$_{\max}$ existing in the efficient Re range is a difficult constraint to meet for the stack, but the constraint on positive excess power was not an issue. The range of increased excess power with reduced ξ over the range of stacks and shapes was at most a 20% increase, and was accompanied by a 20x reduction in charge time. These parameters provide guidance about which parameters to look for in choosing a piezoelectric stack for optimum robot power.

Chapter 5 RIE of *x*-cut Quartz

To make quartz wings for testing, one of the challenges was obtaining thin enough quartz membranes. The thinnest quartz wafer that a vendor was willing to grind for us was 100 μm thick, so I developed a method to thin x-cut quartz from 100 μm to less than 10 μm using reactive ion etching (RIE), and have it still be electrically resistive after metallizing both sides; i.e., free from etch pits.

The drive for this came after fabricated samples were initially electrically shorted from the front to bottom of the quartz. Optical microscope inspection of such a sample before back-metallization is shown in Figure 5-1. The back-lighting (i.e., light traveling through SU-8, gold, chrome, then quartz, as viewed from the quartz-side) reveals that the darker etch pits are actually through-holes, explaining why the sample is shorted from front to back.

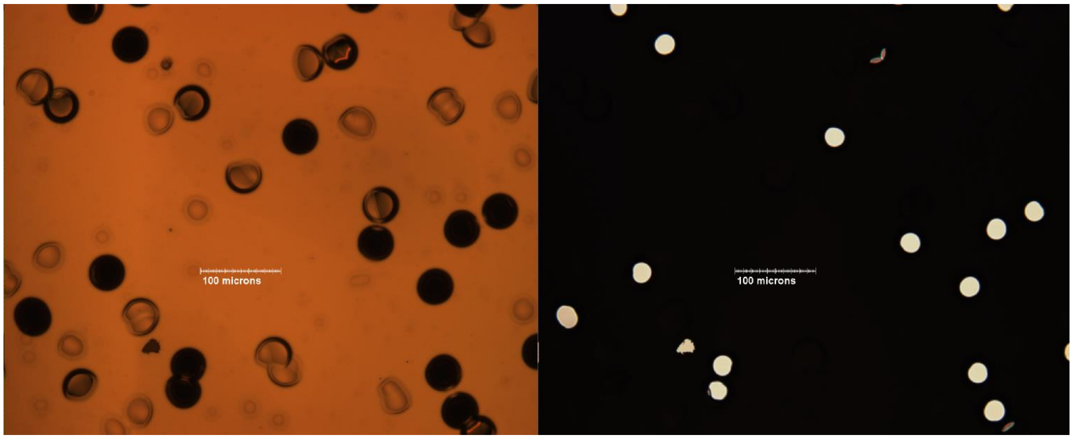


Figure 5-1: Close up on the bottom of QSU8_10 with lighting from the front and back, respectively

On SEM inspection of the etch pits after only a short RIE using SF_6, it became clear that the pits were showing some strange spiral patterns at their base, as shown in Figure 5-2.

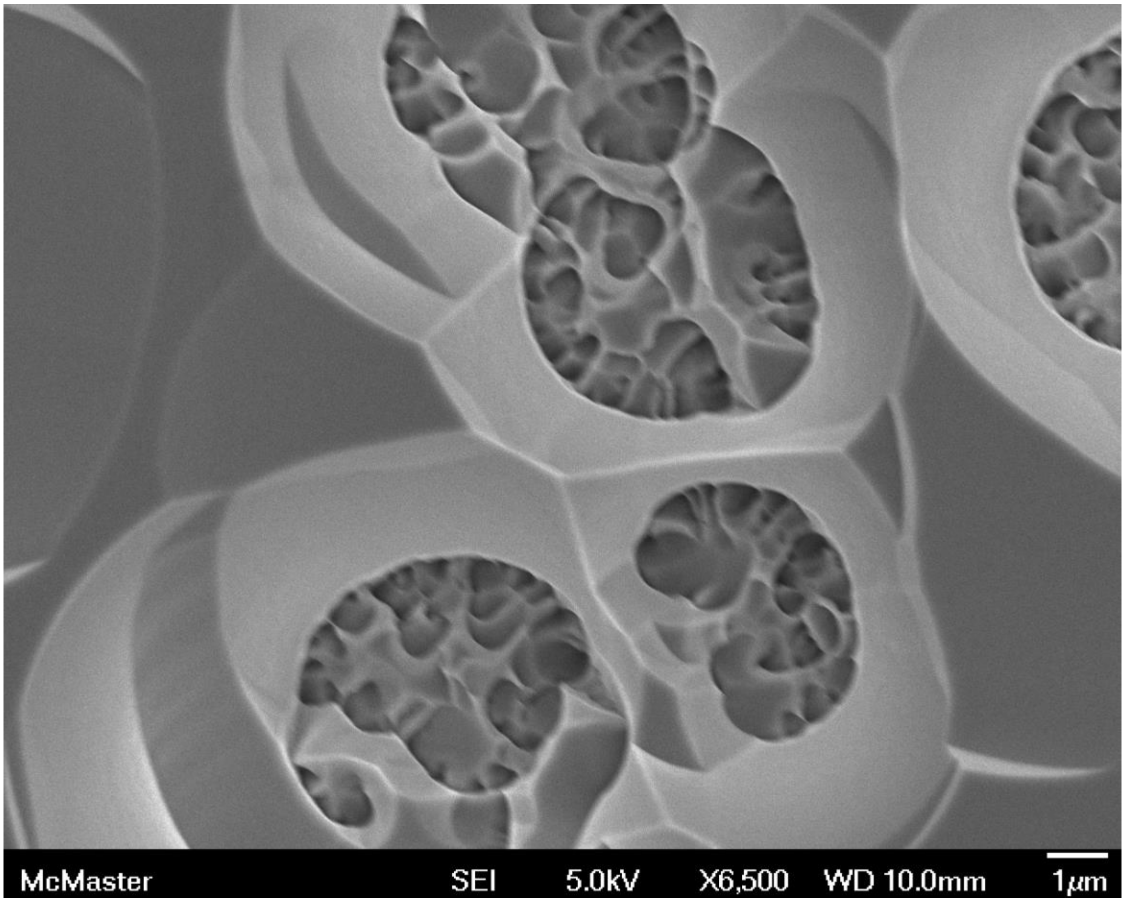


Figure 5-2: Spiral patterns at the bottoms of pits observed after crystal quartz etching with 100% SF_6 for 5.5 hours

In order to learn more about what was causing these pits and how to avoid them, I carried out a detailed study of quartz etching using an STS-320 RIE, varying the shadowmask material and replacing portions of the etch gas with argon. This study was quite informative, and lead to publishing an article ([47]) in Volume 23, Number 11 of the Journal of Micromechanics and Microengineering, Optimum Reactive Ion Etching of x-cut Quartz using SF_6 and Ar, reproduced with permission in this chapter. Part of this work required writing a program to process data output by the optical interferometer to establish a measure for surface roughness and obtain a more accurate measure for height differences. This program appears in Appendix Q.

Section 5.1 Introduction

Single crystal quartz is a piezoelectric material with low-loss elastic and electrical characteristics making it well suited as an electromechanical resonator [48]. Thin x-cut alpha quartz (single-crystal SiO_2 in its natural phase at STP and with the crystal x-axis perpendicular to the surface) is particularly useful for fabricating piezoelectric transducers as quartz couples electric fields in the x-direction to strains in the x- and y-directions [49]. As such, a modest voltage applied across a thin x-dimension will translate into a large electric field and consequently a large displacement in y (if that dimension is much larger, as it can be in a thin x-cut wafer), making x-cut quartz the optimum quartz cut to use for out-of-plane MEMS cantilever actuators. However, it is difficult to chemically thin x-cut quartz since this direction has only one dangling bond compared to two in the z-direction and consequently etches much slower in hydrofluoric acid ([48], [50]), leading to long etch times which make masking difficult and undercuts excessive.

Reactive ion etching (RIE) presents an interesting alternative to chemical thinning of x-cut quartz. RIE combines plasma etching with ion milling: Charged ions physically ablate the surface while free radicals produced by the plasma chemically react with it, forming a volatile species which evaporates in the vacuum of the RIE chamber ([51], [52]). When quartz is thinned by RIE, this combination of ion milling and the high reactivity of fluorine radicals overcomes the chemical stability of the bonds in the x-direction.

Unfortunately, RIE thinning of quartz often induces significant pitting damage on the etched surface which prevents using the thinned membrane as a resonator, due to the formation of through-holes. The problem is compounded by shadow masking (i.e. physically blocking parts of the surface with another material to mask them against etching) and using a stable crystal plane (i.e. x-cut) as the surface rather than amorphous SiO_2. This pitting damage and methods to circumvent it have been studied for amorphous SiO_2 using SF_6 and Ar (or SF_6 and Xe) and the low pressures possible in ICP RIE ([53], [54]). Although developed for amorphous SiO_2, these techniques have been successfully applied to produce resonant structures by thinning y-cut quartz from 100 μm to 18 μm thickness ([55]) and by thinning AT-cut quartz from 25 μm to less than 10 μm ([56]). However, no systematic study of the surface damage produced on single crystal quartz has been made, nor has it been demonstrated whether it is possible to eliminate the damage with standard parallel-plate (non-ICP) RIE. In this work, we systematically investigate RIE pitting damage in shadow-masked x-cut quartz and minimize it using the addition of Ar to SF_6 in a standard, moderate pressure RIE system. Additionally, we examine the role of crystal orientation and shadow mask composition on pitting damage.

Section 5.2 Experimental

Experiments were performed using an STS320 RIE system employing a liquid cooling system. The chamber pressure was fixed at 20 mTorr, and the RF power was fixed at 300 W[4]. This led to a bias voltage which ranged from 650 V to 700 V. The target and the bottom plate were held at 15 ^{o}C.

Etch targets were polished x-cut single crystal quartz wafers provided by LapTech Precision and (where mentioned) polished fused silica (amorphous SiO_2) wafers provided by Valley Design. All targets were segments of 1" discs or smaller samples cleaved from 1" discs. Shadow masks for the argon-concentration trials were 500 μm-thick fused quartz wafers with 8 mm by 2 cm central rectangular windows. Shadow masks for the mask effect trials were 1 cm by 3 cm plates 0.3 to 1 mm thick made of fused silica, alumina, or Ni with 4 mm-diameter holes through their centers. Etching gases were SF_6 and Ar, with total gas flow rate fixed at 18.3 SCCM. All gas percentages are in volume %.

Prior to etching, debris was removed from the surface of the wafers with a dry nitrogen gun, and all etch runs began with a 5 minute oxygen plasma clean (50 W RIE power, 300 mTorr of chamber pressure) to remove residual organic contaminants.

[4] Preliminary trials showed that lower chamber pressures lead to faster and smoother etches while higher powers lead to faster etches without impacting etch smoothness very much. 20 mTorr and 300 W are not an optimum, but the STS-320 could maintain these parameters for (18.3 sccm total gas flow rate) consistently across all etch trials.

In the mask effect trials, wafers were etched in individual runs for 5.5 hours with either no mask or one of the three possible shadow masks using both 100% SF_6 and 87.5% Ar (i.e. 12.5% SF_6 by volume). Etch depth and surface roughness were measured using both a contact and optical profilometer (ZYGO New View). SEM images and EDS spectra were obtained with a JEOL-7000F in secondary electron mode.

Wafers masked only by quartz were etched at several ratios of SF_6 to Ar to study the effects of the Ar concentration on the etch rate and long-etch-time pitting. At 0%, 50%, 75%, and 87.5% Ar these etches were carried out for much longer times to obtain membranes of less than 20 μm thickness. Long etches with 50% Ar were also conducted on fused silica to study the dependence of the pitting and etch rate on crystalline microstructure, and conducted on x-cut quartz masked by Ni to study the long-etch stability of this mask.

Section 5.3 Results and Discussion

Section 5.3.1 Effect of Argon Concentration

Section 5.3.1.1 Effect of Argon Concentration on Etch Rate

The etch rate of x-cut quartz masked with quartz decreased only slightly with increasing Ar concentration until above 90% Ar where the etch rate quickly fell to nearly zero (0.3 μm/hr ± 0.1 μm/hr; see Figure 5-3). Increasing the Ar concentration reduces the SF_6 concentration (at constant total gas flow rate), which lowers the number of reactive species in the plasma and slows down the etch. The surprisingly small initial slope implies that the SF_6 etch is not significantly limited by the concentration of fluorine radicals until the Ar concentration is greater than 90%. This trend is similar to that found by Li *et. al*. for Pyrex ([54]) and by Chen *et. al*. for fused silica ([57]) at the lower pressures of ICP RIE.

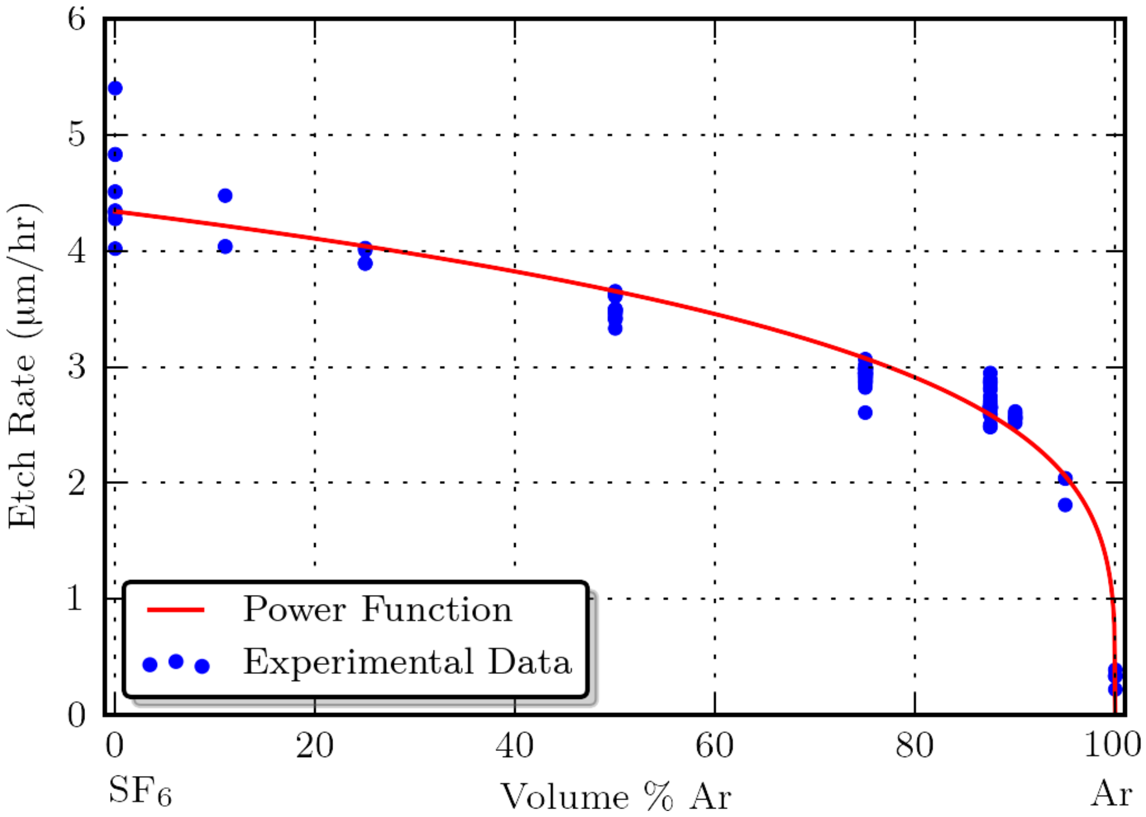


Figure 5-3. Etch rate decay with increasing Ar concentration relative to SF6. Total gas flow rate, pressure, and power are fixed. The etch is fastest with 100% SF6 and slows only moderately with increased Ar concentration until it is above 90%. This suggests that for stable surfaces like those of x-cut quartz, increasing the ion milling at the expense of free radicals gives a relatively minor reduction in the etch rate as long as some free radicals are still present. Phenomenologically, the trend is well described by a power law function[5] (as shown).

Section 5.3.1.2 Effect of Argon Concentration on Surface Quality

With SF_6 alone as the process gas, pit depth accelerates as the etch progresses, potentially leading to through-holes in the membrane. This is believed [58] to be due to the plasma induced surface damage (PISD) mechanism: Material ablated from the mask or the substrate redeposits on the surface of the sample, perhaps after reacting with the plasma

[5] $Rate = 4.3(1-\mathrm{ArRatio})^{0.25}$

species, forming a small local shadow mask known as a micromask. Until they are removed by etching, ablation, or possibly evaporation, these micromasks shield the substrate underneath them and deflect incoming ions and radicals at an angle into the surrounding surface. Due to x-cut quartz having a slow directional etch rate, these deflected species can preferentially etch faster etching crystallographic surfaces, particularly the {122} family [59], resulting in pit formation which accelerates once facets are exposed. Even after the micromask is removed, the exposed facets continue to etch more quickly than the surrounding surface. This results in severe pitting as shown in Figure 5-4.

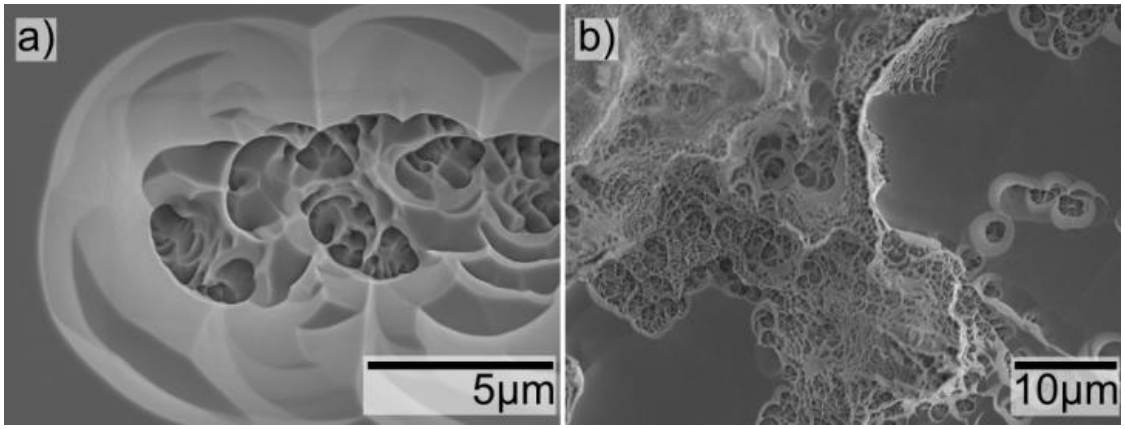


Figure 5-4. Deep pitting in quartz etched using 0% Ar (SF_6 only) after only 5.5 hr (26.5 □m etch depth), a) near the center and b) near the edge of the etch mask.

After etching 72 μm of a 100 μm wafer, these pits are deep enough that they break through, providing conduction paths from the metal contact on each side of the membrane, rendering it inoperable as a piezoelectric device.

PISD pitting is overcome by adding sufficient Ar to the process gas mixture, as shown in Figure 5-5. While a small addition of Ar reduces the pitting at all etch times, we found that a much more significant improvement occurs at 75% Ar and above: At these concentrations, the pits no longer showed a noticeable increase in depth or areal density with increased etch time, as shown in Figure 5-6. The addition of Ar to the RIE process enhances the physical sputtering action, cleaning contaminants off the surface, and destroying the faster etching crystal facets. This allowed fabrication of very thin insulating quartz membranes with high aspect ratios (7 μm x 6 mm x 20 mm).

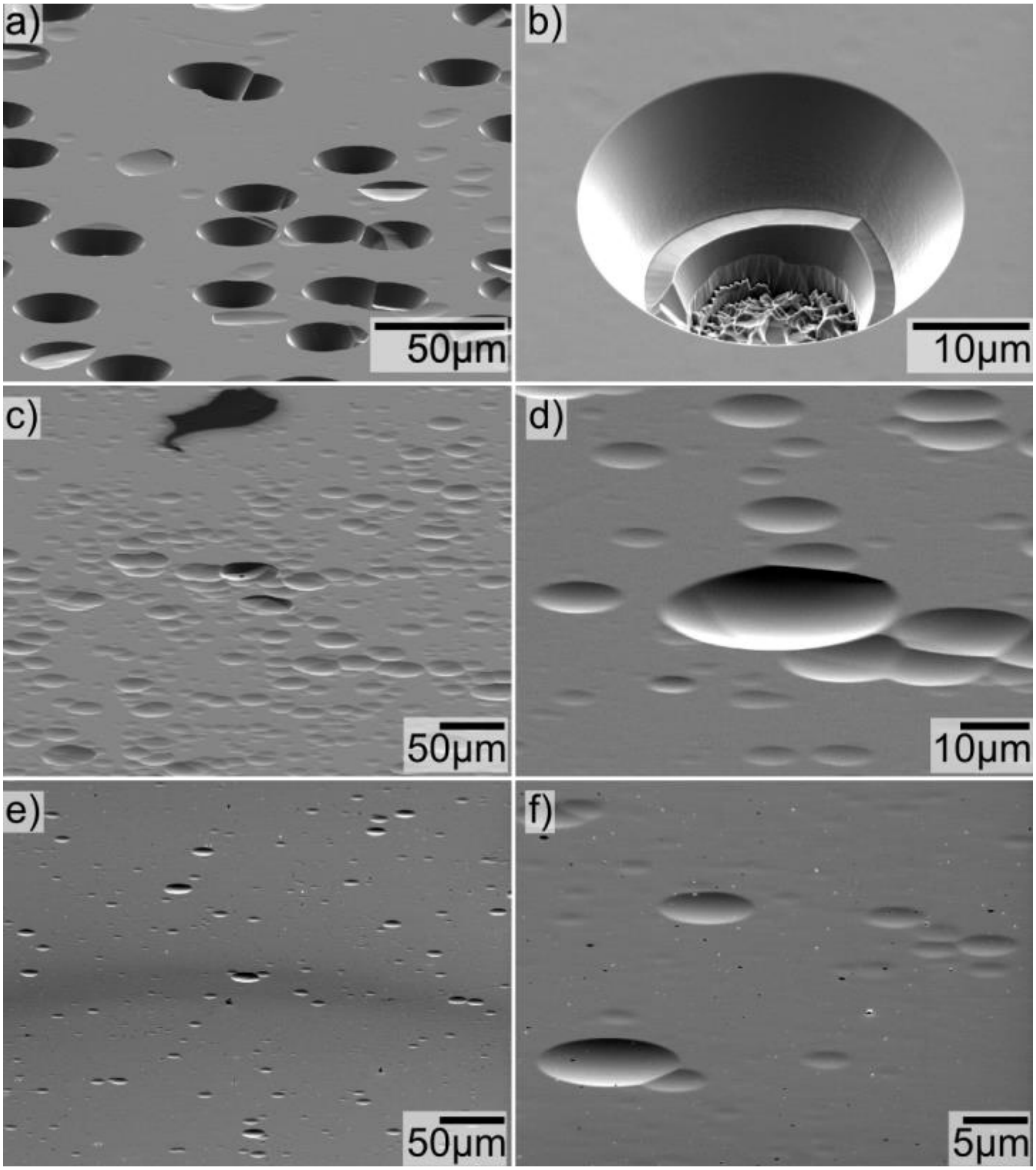


Figure 5-5. Reduced pitting after long etch times with increasing Ar%: a-b) 50% Ar, 21.25 hr c-d) 75% Ar, 29.73 hr, e-f) 87.5% Ar, 33 hr. With 50% Ar, pit depth increases with etch time and breaks through the thin membrane resonator as shown in b) (72.2 μm etched out of a 100 μm wafer). 75% or more Ar did not show a pit depth increase as the etch progressed and 87.5% Ar showed even lower pit depth and density.

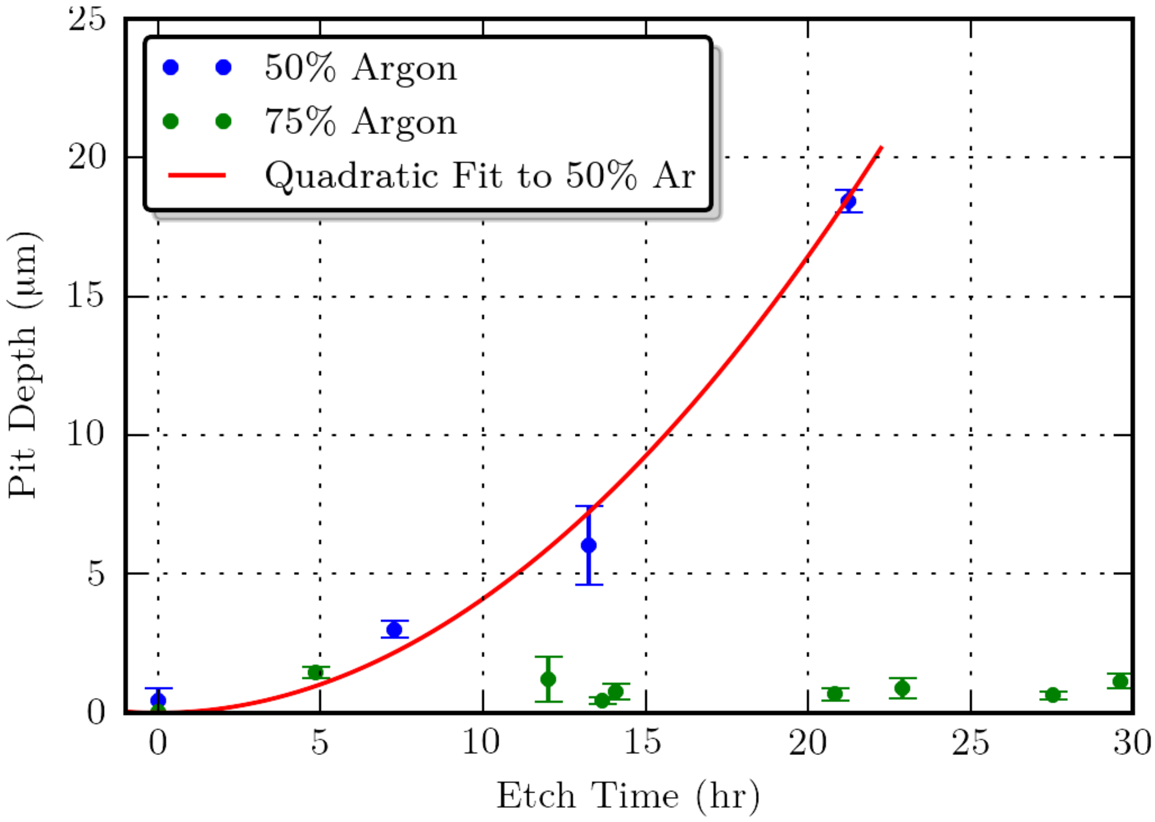


Figure 5-6. Pit depth *vs.* etch time for 50% and 75% Ar. This plot illustrates that pit depth increased as the etch progressed for 50% Ar, but not for 75% Ar.

Section 5.3.2 Effect of Crystalline Microstructure

Section 5.3.2.1 3.2.1. Comparing to Fused Quartz Etching

The results of a 21.25 hr 50% Ar etch of *fused* silica (amorphous SiO_2) are shown in Figure 5-7. The pitting and jagged-angle spiral pit bottoms observed in figure 3 for crystal quartz etched with 50% Ar are much reduced for fused silica. This supports the hypothesis that the spiraling pits observed in single crystal quartz are due to the crystal structure and the anisotropy it entails. Fused silica etched moderately faster than x-cut quartz under the same etch conditions: 4.6 μm/hr *vs*. 3.4 μm/hr.

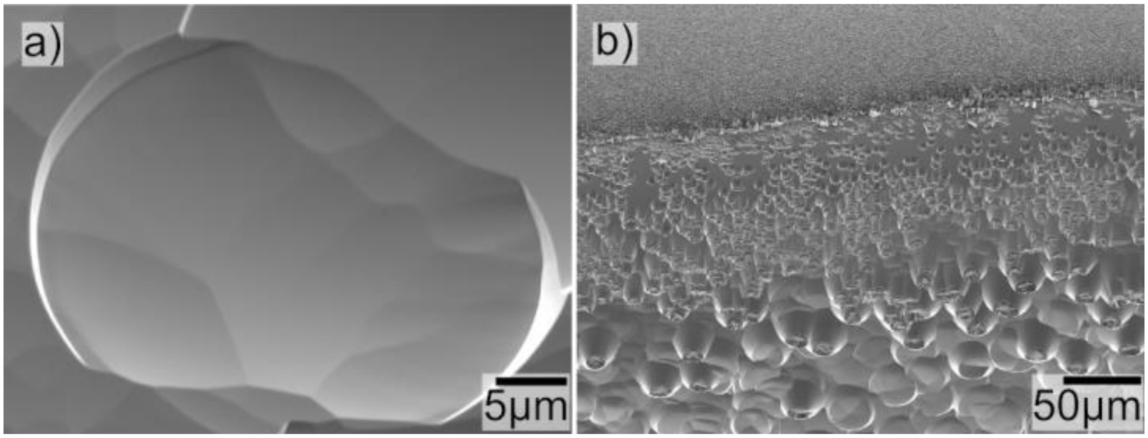


Figure 5-7. Results of etching fused silica with 50% Ar (50% SF_6) for 21.25 hrs, depth of 97.4 μm, a) typical center pit and b) increased pitting near the edge of the shadow mask. The smoother angles observed, especially at the bottom of the pits, supports the hypothesis that the crystal direction dependence of the etch rate is responsible for the sharp-angle pitting when etching x-cut quartz.

Section 5.3.2.2 Micromasking vs. Crystal Defects

We also explored the possibility that the pitting is due to crystal defects rather than micromasking exposing high-index planes. In contrast to micromasking, if the pitting is due to crystal defects, then proximity to a wall and sample cleanliness should not appreciably effect the pitting. To test this, we cleaned one area of a sample by etching it for 6 hr with 100% Ar while masking the other area with fused quartz. Following this, the mask was removed and both areas were subjected to 21.25 hr of 50% Ar etching. The results are shown in Figure 5-8.

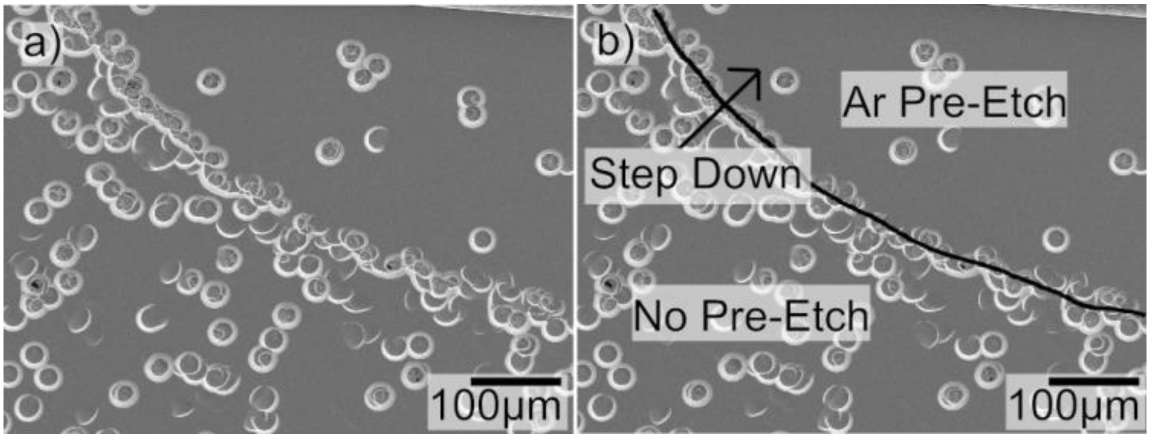


Figure 5-8. Reduced (but not eliminated) pitting from an extra Ar pre-etch and a quartz ridge transformed into a curve of densely-packed pitting, a) clear and b) annotated.

The Ar pre-etched area resulted in locally less pitting from the SF_6-containing etch that followed. Since both regions were subjected to identical SF_6-Ar etching for the rest of the run, this is inconsistent with the pitting explanation being due to crystal defects. As the etch involving SF_6 begins, if the crystal surface is not etched uniformly because of dirt on the surface, this non-uniform etching could locally expose high-index planes as the material is removed, leading to pit formation. Further note in Figure 5-8 that the step down (by 2.2 μm) to the area with 6 hr of extra Ar-only etching has, after etching involving SF_6, turned from a step into a curve of densely-packed etch pits. Since this pit formation occurred when there was not actually an etch mask adjacent to this part to increase micromasking, it suggests an additional source of increased etching near walls: walls of the quartz have faster-etching planes exposed. This promotes pit formation and is consistent with the structures observed near masked walls, such as Figure 5-4b.

Section 5.3.3 Effect of Masking Material

Section 5.3.3.1 Effect of Masking Material on Etch Rate

Figure 5-9 compares the etch rates for quartz using different shadow mask materials, for both 100% SF_6 and 87.5% Ar etch gases. The etch rate for 100% SF_6 was significantly lower using an alumina mask than with a Ni mask, fused silica mask, or no mask. Alumina, while resistant to RIE thinning, is still sputtered somewhat and redeposits on the target material forming a layer which inhibits etching. This effect was alleviated for 87.5% Ar. Fused silica and Ni did not noticeably reduce the etch rate beyond that of no shadow mask.

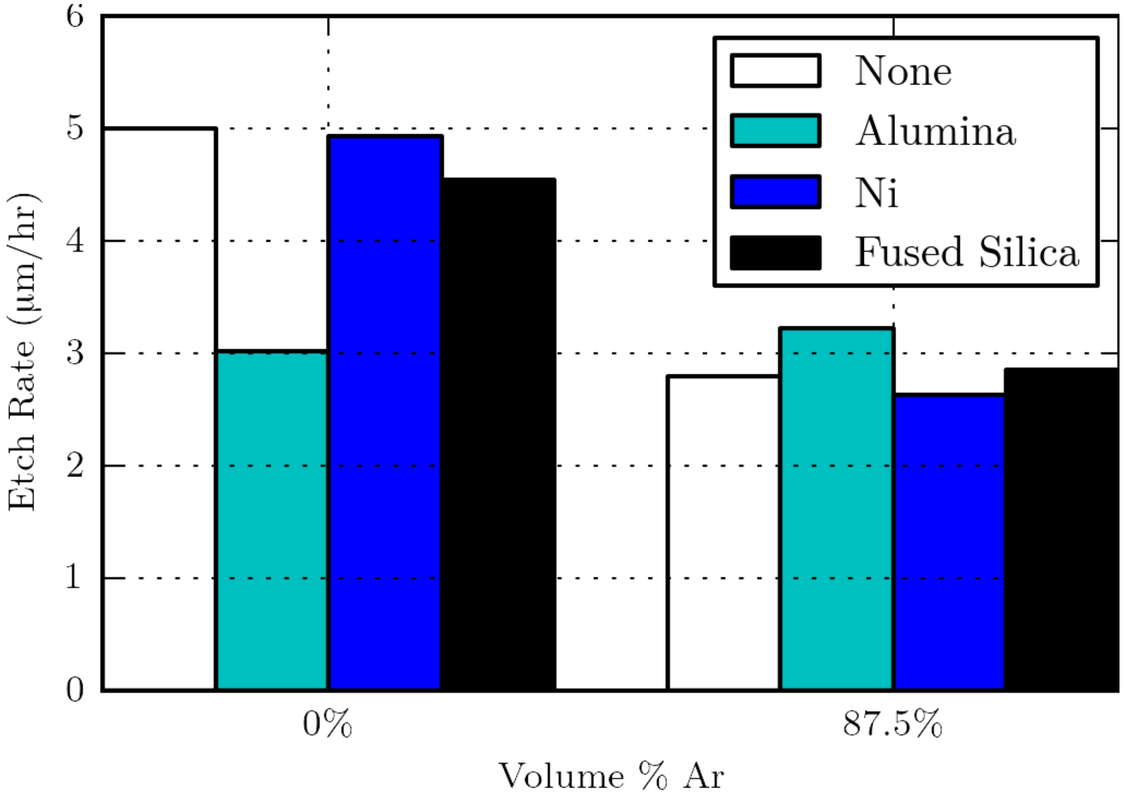


Figure 5-9: Etch rate for 100% SF_6 and 87.5% Ar using different shadow mask materials, measured after 5.5 hrs of etching. The etch rate for 100% SF_6 was not affected by shadow masking with fused silica or Ni, but was significantly slowed when masking with alumina. This was not observed when sufficient Ar was present, supporting the hypothesis that the reduction in etch rate for alumina is due

to redeposition of alumina on the surface of the quartz that is alleviated by increased ion milling with Ar.

Section 5.3.3.2 Effect of Mask Edges

Pitting showed an increased density immediately next to where the mask was placed. This increase, which typically extended only tens of microns out from the walls, is believed to be due to the increase in high-index planes of quartz that arrive next to the wall as the etch progresses due to the nonuniform direction of incident etch species (see Figure 5-8, and Figure 5-4b). Surface conditions in the majority of the etched region were uniform, and largely independent of distance from the mask. The residue from the Ni-masked 100% SF_6 etch dropped off 10 mm from the mask edge, after which point the surface appeared gradually closer to the quartz or no-mask case. Likely this effect is strongly dependent on chamber pressure. Unfortunately, the 100% SF_6 alumina-masked sample only had 6 mm clearance from the mask to the sample edge and no drop in residue was observed over this distance.

Section 5.3.3.3 Effect of Masking Material on Surface Quality

The surfaces after masking with alumina, Ni, and fused silica, are shown in Figure 5-10 and Figure 5-11, with respect to a control sample having no mask. Alumina masking led to rough surfaces with no Ar, and even with Ar present led to some surface damage after extended etch times. Due to the resulting morphology, shadow masking with alumina using only SF_6 etching has potential application as an antireflective coating [60]. EDS measurement in SEM indicates the formation of aluminum fluoride on the surface of the masked quartz.

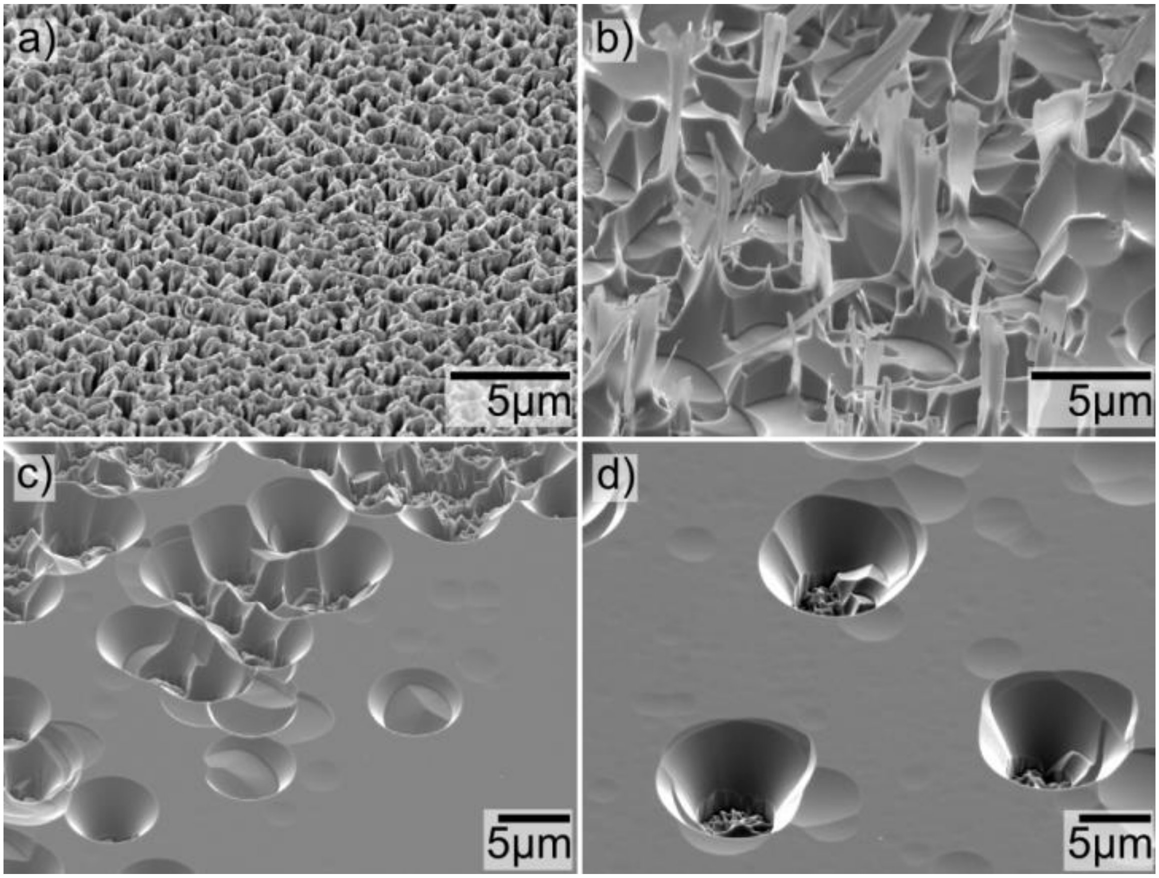


Figure 5-10: Surface of x-cut quartz after etching for 5.5 hrs with 100% SF_6 while shadow masked with a) alumina, b) Ni, c) fused silica, and d) no mask. The quartz surface is not only etched far more slowly with alumina as the shadow mask, but also greatly damaged. Ni masking gave more pitting than silica masking and left residue on the surface. Pitting with fused silica masking is similar to no masking.

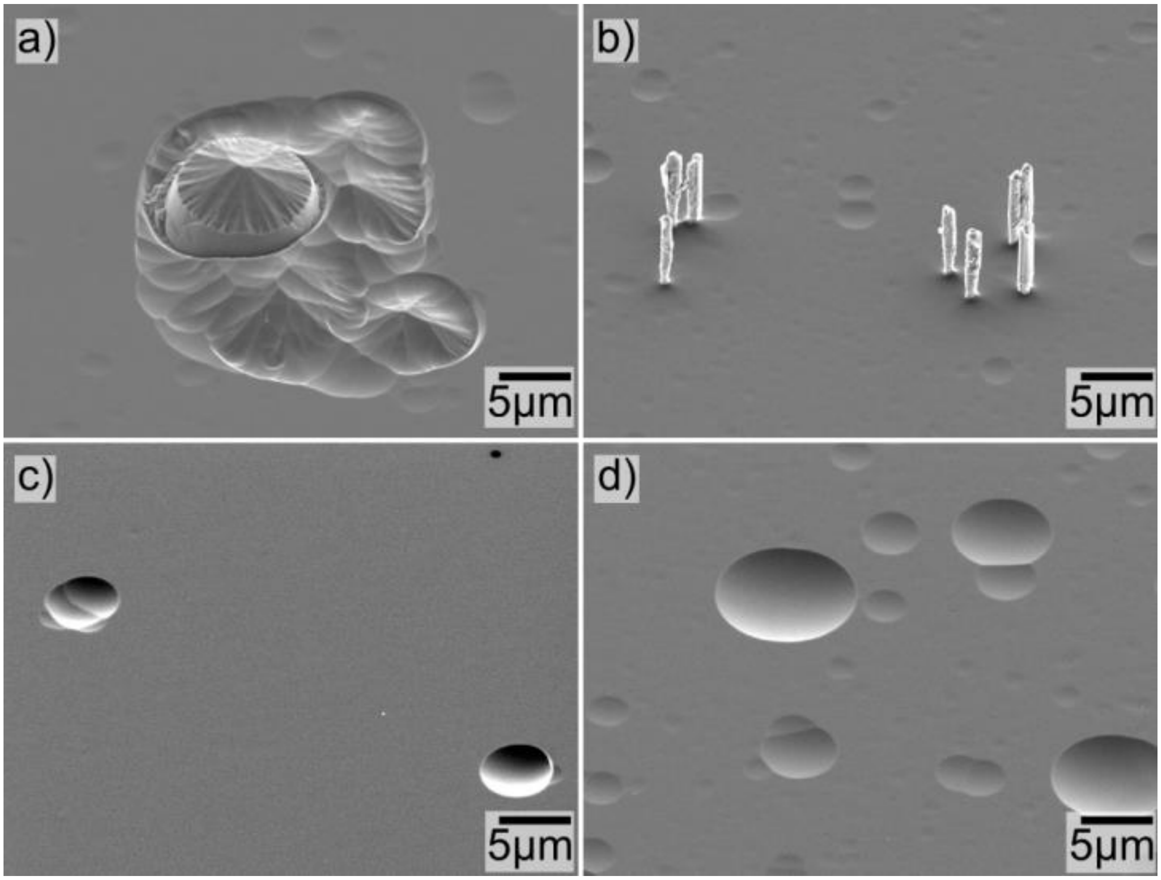


Figure 5-11: Surface of x-cut quartz after etching for 5.5 hrs with 87.5% Ar while shadow masked with a) alumina, b) Ni, c) fused silica, and d) no mask. The shadow masking material has a much smaller effect on surface quality when Ar is present in the etch gas. The surface damage with an alumina mask is greatly reduced by the introduction of Ar gas. Initial pitting is similar for Ni and quartz masks but surface damage is still present for non-quartz masking. Ni masking resulted in hollow column-shaped surface structures.

The standard shadow mask for quartz RIE, Ni, left cleaner surfaces than did alumina but not as clean as masking with fused silica. This is expected because masking with fused silica introduces no additional surface chemistry (the substrate and mask are both SiO_2), and so presents no additional possibility for non-volatile compounds to micromask the surface. Without Ar in the gas mixture, the quartz surface after masking with Ni was quite rough (see Figure 5-10b). We even observe what appear to be shards of metal on the surface. EDS confirmed the presence of Ni and nickel fluoride on the surface after etching.

While Ni and silica masking of quartz produced similar surfaces after 5.5 hours etching with 87.5% Ar (Figure 5-11b and c), silica masking proved superior for long etch times. Figure 5-12a shows that x-cut quartz masked with Ni and etched using 87.5% Ar for 34.5 hrs (96.5 μm depth) left a surface which was quite rough and covered with hollow pillar-shaped structures similar to those observed in Figure 5-11b after ecthing for 5.5 hrs. The membrane produced after this etch (~7 μm thick quartz) was electricaly shorted from one side to the other through its bulk, indicating the surface roughness causes sufficient pitting to create conduction paths through the wafer. Adding Ar does not eliminate the pillars, but changes their shape: The result of etching using a Ni shadow mask with 93.75% Ar for 22.8 hrs (55.5 μm depth) is shown in Figure 5-12b. Large conical-shaped structures appeared on the surface composed of quartz and non-volatile nickel fluoride. Insufficient SF_6 may hinder the lateral etching of these columns leading to their wide base compared to the pillars observed with 87.5% Ar. This effect was not observed with silica masking, and suggests that Ni is not an appropriate shadow mask for non-ICP RIE where smooth surfaces are required after deep etching. Despite the appearance of these structures, the etch rate using Ni masking did not mesuarably decline over the course of the etches, indicating that Ni is a suitable shadow mask even for non-ICP RIE where a through-etch is required.

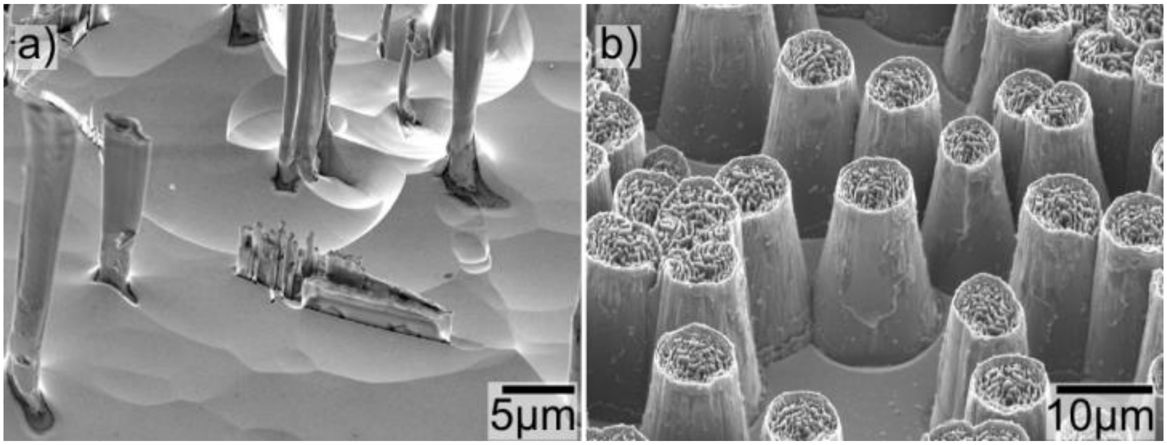


Figure 5-12. Structures observed on the surface for deep etching x-cut quartz with Ni shadow masking using a) 87.5% Ar and b) 93.75% Ar. Lowering the SF_6 concentration reduces the lateral etch rate and changes the structures from a) hollow pillars to b) conical-shaped structures.

Section 5.4 Conclusions

Thinning x-cut quartz via RIE gives the fastest etch rates with 100% SF_6 as the process gas, however the smoothest surfaces (particularly for deep etches) are obtained with Ar making up 75% of the gas mixture with only a 30% reduction in etch speed. Ni is a better shadow mask for quartz than is alumina for both etch speed and final surface quality, but fused silica is better still (although it is etched just as fast as the substrate). After long etch times, Ni masking leads to significant surface damage, while smooth surfaces are obtained with silica masking. Although etch rates for ICP RIE systems [61] are 30 times faster than presently reported, we have demonstrated that it is also possible to reach impressive etch depths and create thin electrically-insulating membrane structures from x-cut quartz using more readily available non-ICP RIE systems, and we have optimized the process parameters for doing so.

Chapter 6 Creating the Wings

As an initial proof of concept, I opted to work with a quartz:SU-8 wing design. SU-8 is a UV-cured epoxy, which can be spin-coated directly on to quartz, patterned, and used as a structural material with a built-in tensile strain when deposited on quartz. Although the optimization routine of Chapter 4 indicates that quartz:Si leads to superior wing performance, using quartz:SU-8 involved simpler fabrication because depositing SU-8 is significantly easier than the wafer bonding or polysilicon deposition required for quartz:Si wings.

While robot wings themselves will be attached only to a fuselage and battery, for testing purposes the wings are attached to a wafer and printed circuit board (PCB). This "test structure" design has some differences from the standalone wings. For one, it possesses a "bridge" section clamping the centre of the wings, which modestly changes the resonant boundary conditions. Also, it has only a narrow hole for air to pass through, and so can alter the fluid mechanics compared to free wings.

Two pairs of fabricated wings and attached test structure are shown in Figure 6-1. There were several working test structures with various changes in the fabrication to give different layer thicknesses and latent strains. In this thesis, I only report the force on QSU8_17a, as it was the most extensively tested. As such, I will focus exclusively on QSU8_17a's fabrication recipe.

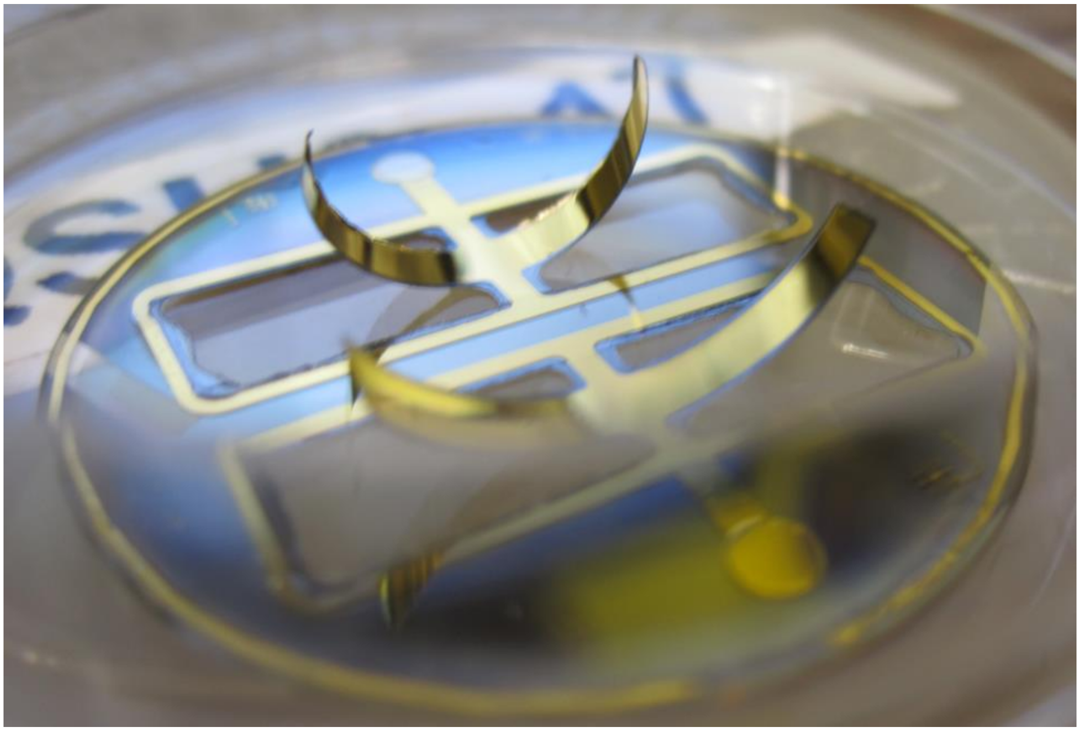


Figure 6-1: QSU8_17 in August 2011. Back: QSU8_17a, front: QSU8_17b.

The stack structure for QSU8_17a, is given in Figure 6-2. The SU-8 has a latent tensile stress in it compared to the rest of the structure as a result of the polymerization process and its higher thermal expansion coefficient than quartz.

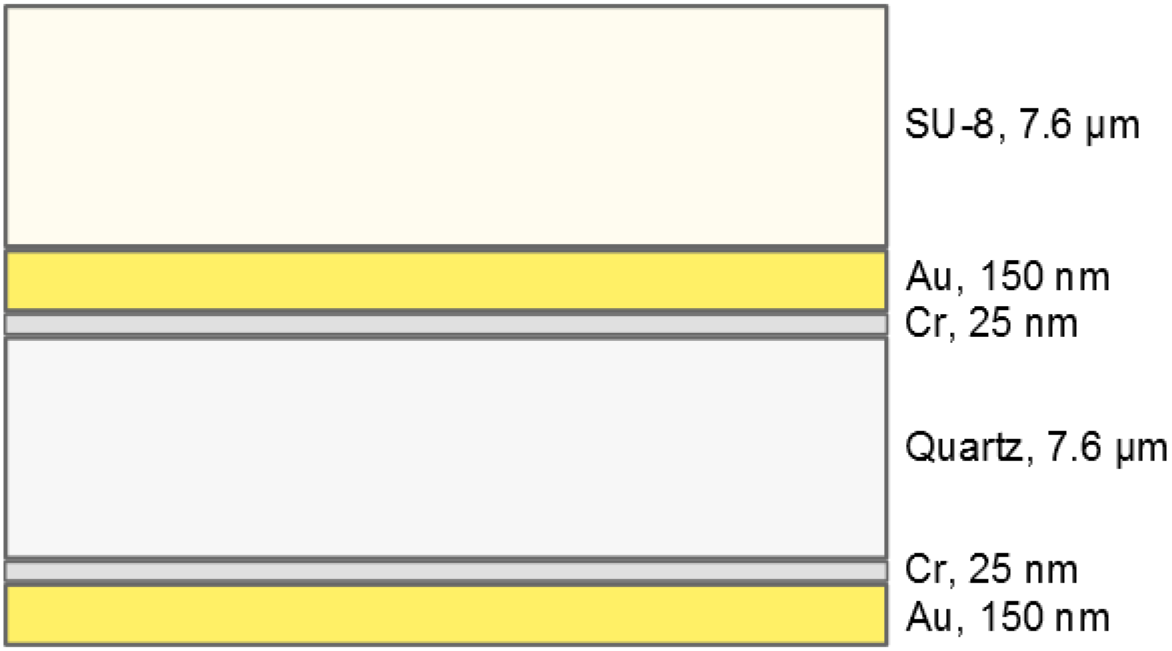


Figure 6-2: Layers for QSU8_17a.

Using SU-8 as the passive layer for quartz wings, the fabrication is broken into 5 phases:

1. Patterning the metal contacts onto the quartz top side. This will be the electrical contact in the middle of the quartz and SU-8.
2. Patterning the SU-8 on the top of the quartz. SU-8 is the passive structural layer for the unimorph.
3. Thinning the quartz from the bottom side with RIE. Only in a section around the wings, in order for the wafer to maintain structural integrity for ease of handling.
4. Metallizing the back side of the quartz. This will be the other electrical contact for the quartz.
5. RIE releasing the wings. This step defines the wing shape. The strain difference between quartz and SU-8 causes the wings to lift after the RIE mask is removed.

Section 6.1 Fabrication Steps Explained

The detailed steps in this fabrication recipe are presented in Appendix R. Final thicknesses of the metal layers were determined using a quartz crystal microbalance during deposition, and final thickness for the quartz and SU-8 layers were determined using optical profilometry. The quartz crystal microbalance is part of the e-beam evaporation metallizer in the Centre for Emerging Device Technologies (CEDT)'s clean room at McMaster University. I wrote custom software to analyze the optical profilometry measurements, discussed in Appendix S. Following is the recipe with a brief explanation for each of the 5 fabrication steps.

Section 6.1.1 Pattern the Metal Contacts onto the Top Side of the Quartz

After cleaning the quartz wafer in solvents, deposit and pattern positive photoresist (1808) using a photomask (all photomasks were ordered from Fine Line Imaging), the dimensions of which are shown in Figure 6-3, then evaporate a thin layer of chromium onto the quartz surface followed by gold. The gold is the main conductor of electricity, and the chromium is necessary to adhere gold to the quartz. Dip in acetone to remove the gold on top of the 1808, then remove any residual acetone using methanol. The result is a metallized surface like that shown in Figure 6-5, although without SU-8 on it just yet.

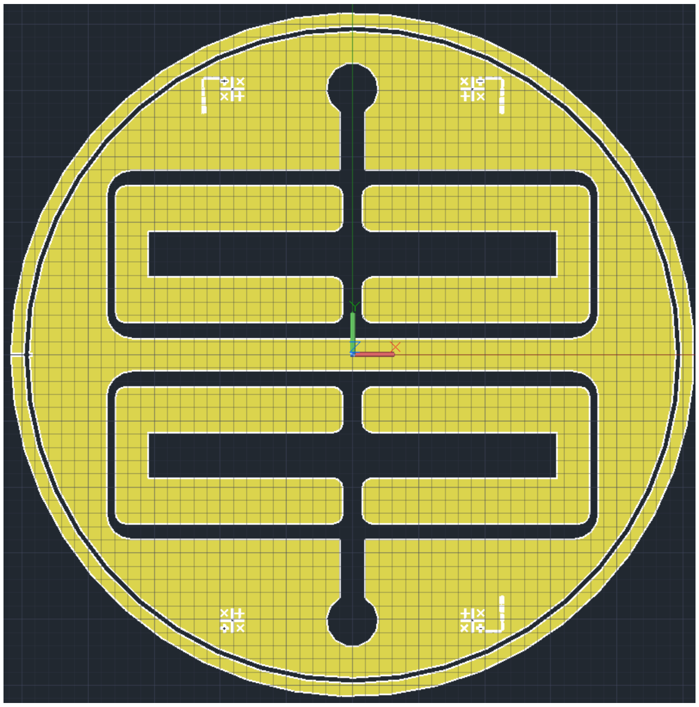


Figure 6-3: Electrical contact mask

Section 6.1.2 Pattern the SU-8 on the Top of the Quartz

Spin-coat SU-8 2007 at 3000 rpm, then expose using the mask of Figure 6-4. SU-8 is a negative photoresist (technically it is a UV-cured epoxy, but that means it is exposed like a negative photoresist), and so the areas of SU-8 where the mask is blank will be cross-link polymerized on UV exposure, and so will remain on the surface after developing.

The most important feature of this mask is that it leaves SU-8 on the wings, but does not leave SU-8 on the electrical contact pads. This mask also removes the SU-8 immediately around the wings, but since SU-8 is easy to remove in RIE, it just as well could have left the SU-8 around the wings in place, relying solely on the RIE to define where the wings are located. In an industrial process to make these kinds of wings, that would be unnecessary extra RIE time, but for this process it would have been better to leave it in place to avoid problems associated with a slightly misaligned RIE mask, such as nonuniform SU-8 coverage on the wing. Following the SU-8 deposition, the surface appears as shown in Figure 6-5. SU-8, like quartz, is transparent to visible light, but close inspection reveals its outline extending slightly past the gold around the edges of the wings.

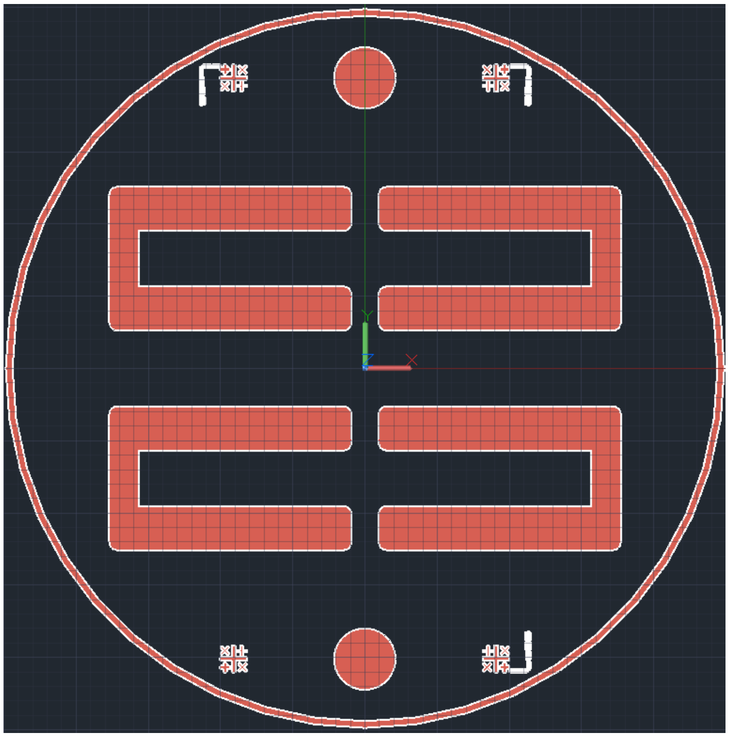


Figure 6-4: Photomask for SU-8 deposition

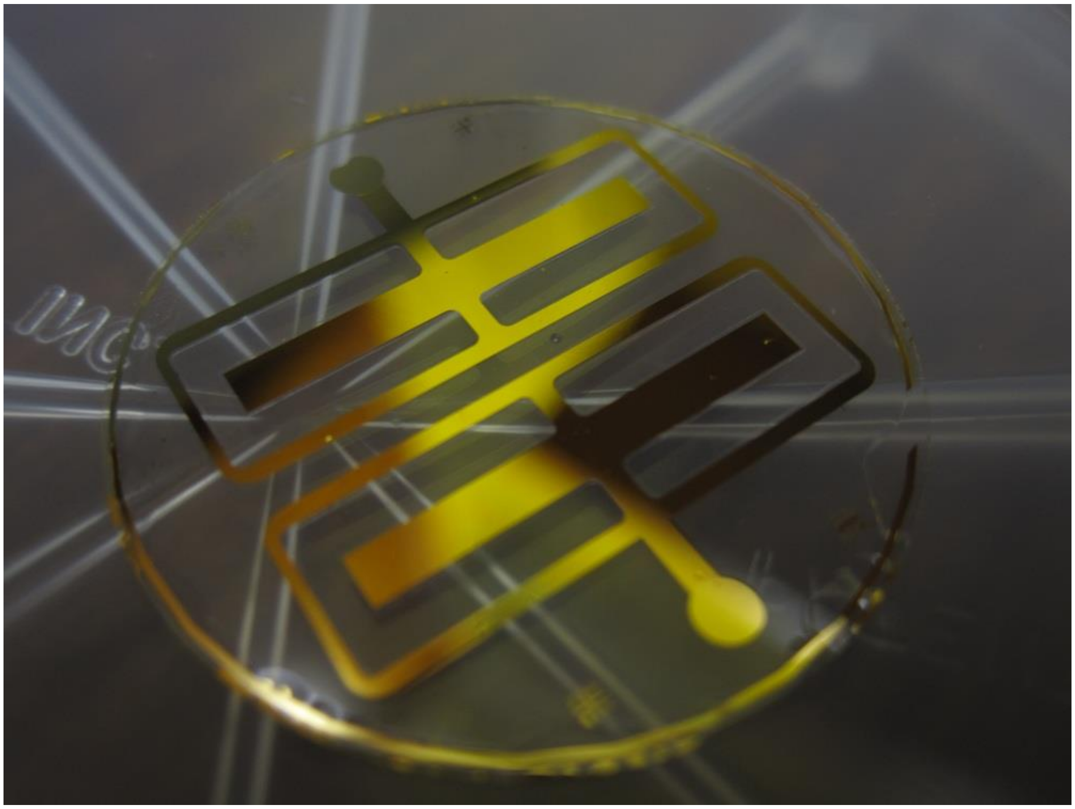


Figure 6-5: Surface after front-metallization and SU-8 deposition

Section 6.1.3 Thin the Quartz from the Bottom Side using RIE

With the top processing complete, flip the wafer and place it into an STS-320 RIE under a quartz shadow mask, and etch for 37.6 hours using 90% Ar, 10% SF_6 by volume at a chamber pressure of 13-15 mTorr, 300 W power, in order to maintain a smooth surface, as explained in Chapter 5. The quartz shadow mask is shown in Figure 6-6. This shadow mask was produced by hand, using a rotary diamond tool (with water running onto the surface to minimize breathing quartz dust). This leads to the rough edges shown for the mask, but the edge quality is unimportant because this is not a through-etch and so the edge does not define much about the final dimensions.

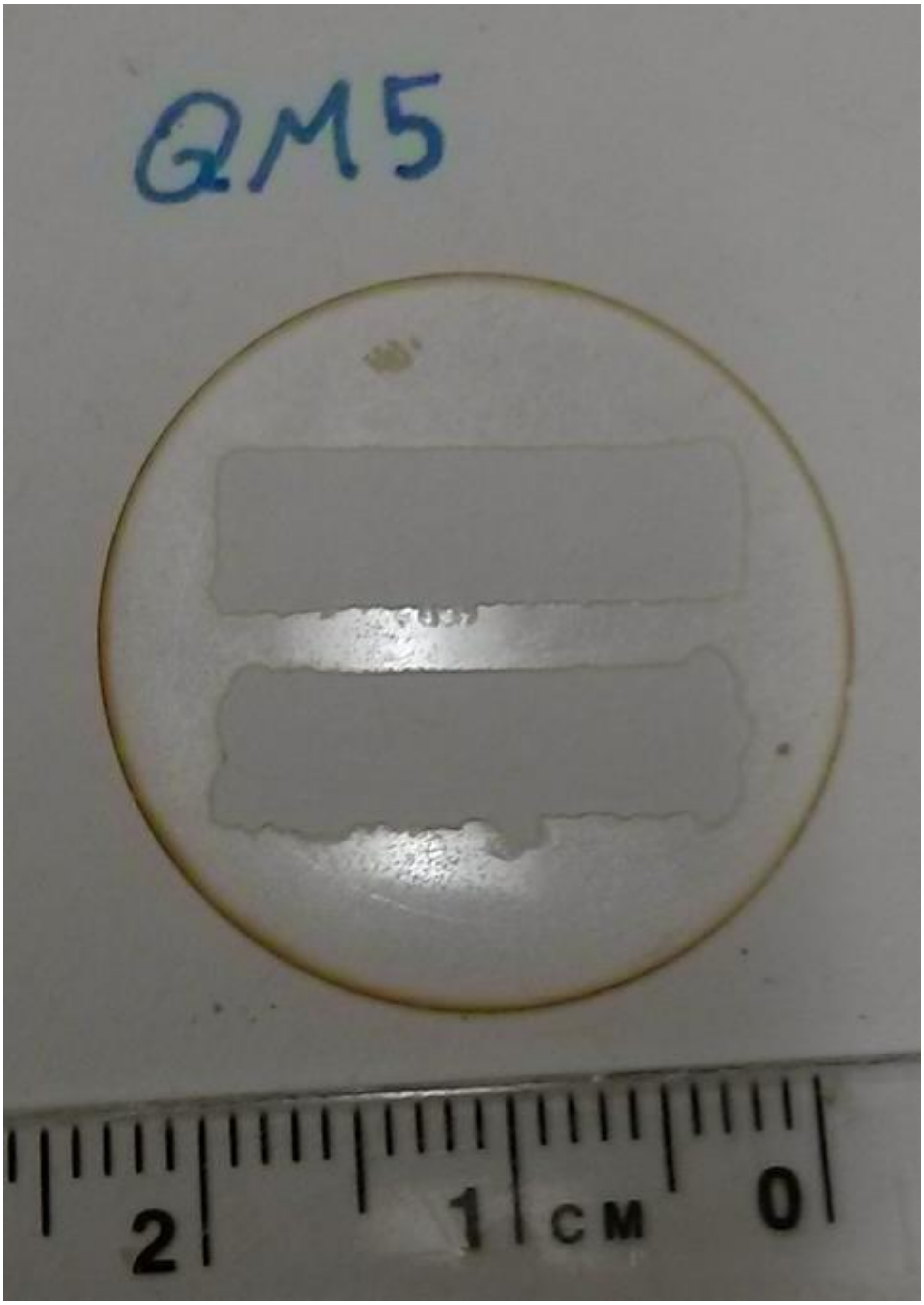


Figure 6-6: QM5: The quartz mask used for back RIE of QSU8_17, after its retirement

This RIE thins down the quartz by 96 μm, from an initial 104 μm to only 7.6 μm thickness (determined by optical profilometry). Following the thinning, the sample appears as shown in Figure 6-7.

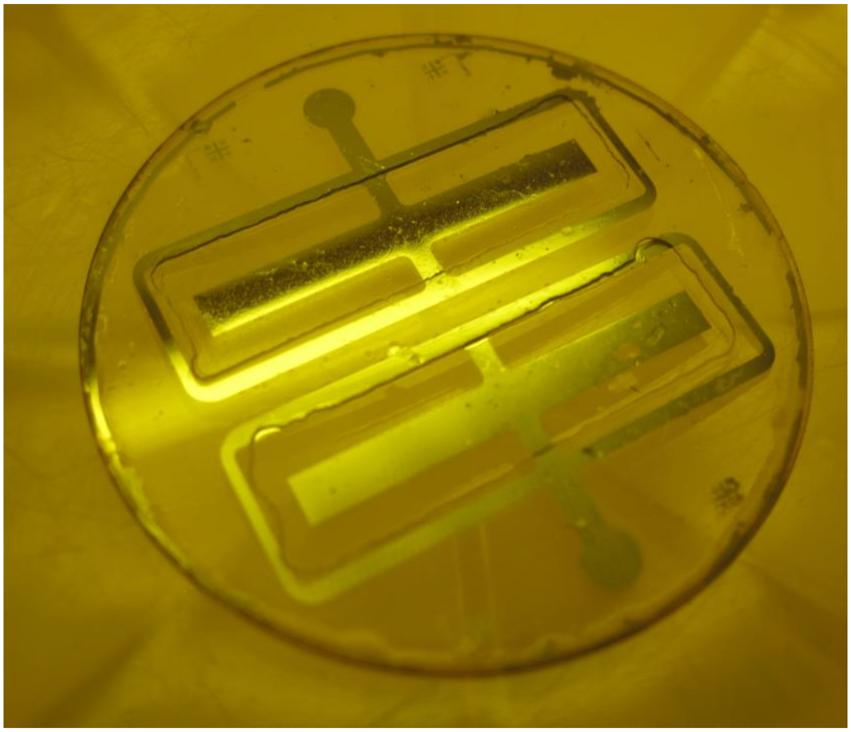


Figure 6-7: QSU8_17 after RIE back-thinning

Section 6.1.4 Metallize the Back Side of the Quartz

With the back of the quartz thinned, it can now be metalized to allow electrical contact to the back of the wings. For this metal coating, it is critical that the un-thinned back of the wafer ends up electrically connected to the thinned part. Therefore, rather than using the low-pressure e-beam evaporation metallization used for the front side, this step uses a relatively high-pressure metal sputtering (5e-3 Torr vs. the 3e-8 Torr of the e-beam system). The higher chamber pressure of the sputtering system leads to a non-directional metal deposition that coats the somewhat-sharp sidewalls of the etched area, connecting them to the unetched area. The result is shown in Figure 6-8.

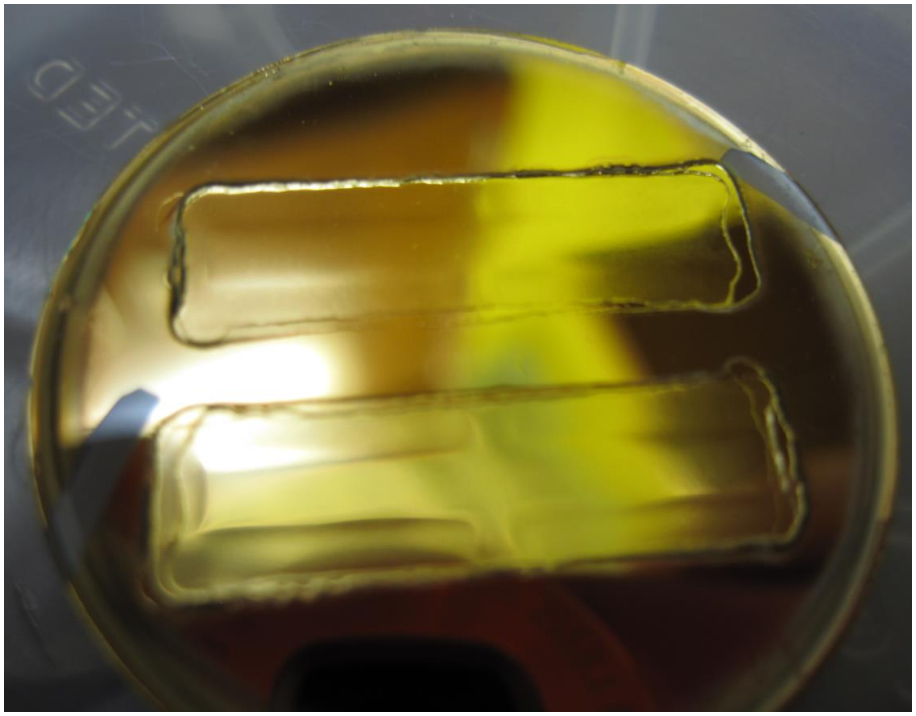


Figure 6-8: Back-side of QSU8_17 after back metallization but before RIE wing-release

Section 6.1.5 RIE Release the Wings

The final step is to release the wings by RIE using a shadow mask made of Ni. The final lateral wing dimensions are determined by this shadow mask, a section of which is shown in Figure 6-10, and dimensioned in Figure 6-9. This mask was fabricated using water jet machining. The gold trail in Figure 6-9 shows the path of the waterjet cutter, which has a 0.5 mm nozzle radius. When the waterjet starts, the initial blast is at a higher pressure and leaves a wider hole than afterward. Because of this, the trail is designed to start at an exterior edge where the dimensions are not critical. The water jet radius is equal to the radius of the resulting fillets on the bridge: 0.5 mm.

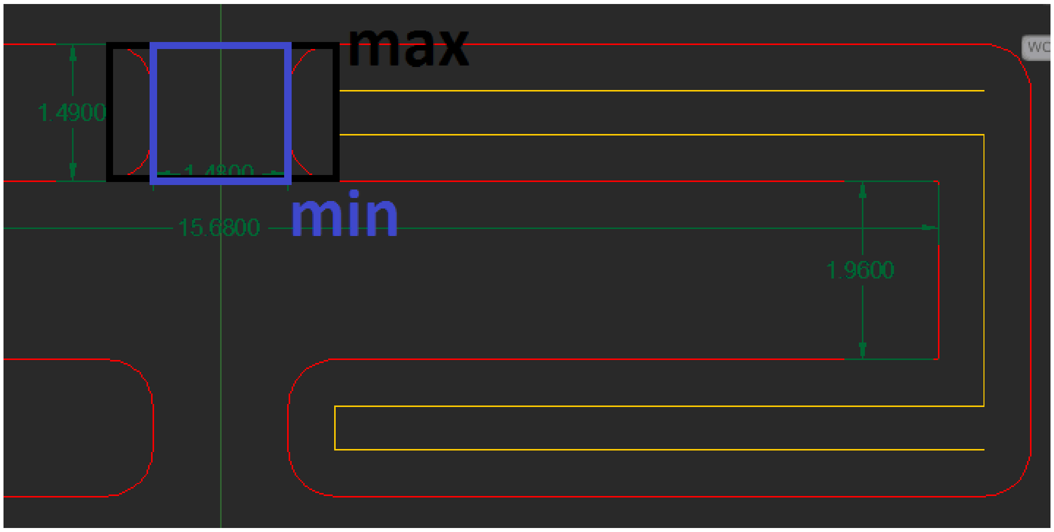


Figure 6-9: Section of RIE release mask showing dimensions in mm

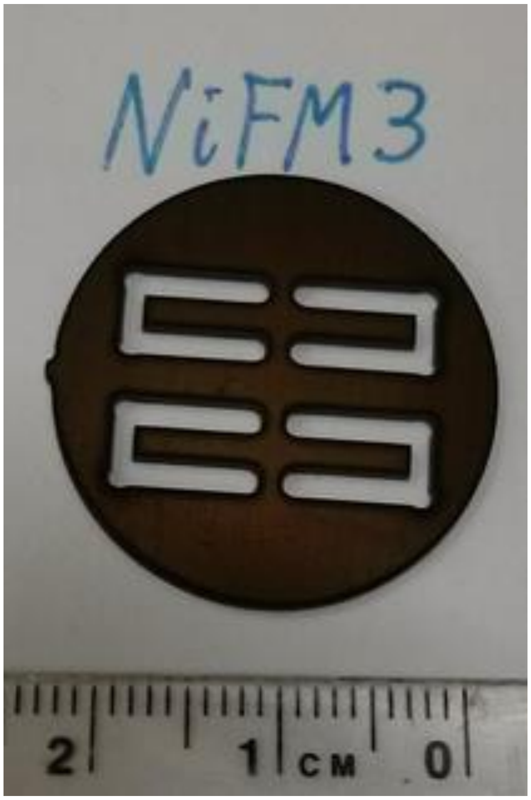


Figure 6-10: NiFM3, the Nickel Front Mask used for RIE wing release of QSU8_17

The shadow mask shown in Figure 6-10 is after use in the RIE, which causes the initially shiny Ni surface to blacken. Energy-dispersive X-ray spectroscopy (EDS) results suggest that the blackness is due to nickel-fluoride.

Following RIE release, the latent tensile stress in the SU-8 layer partially relaxes by curving the wings out of plane as soon as the shadow mask is removed. The resulting structure is shown in Figure 6-11.

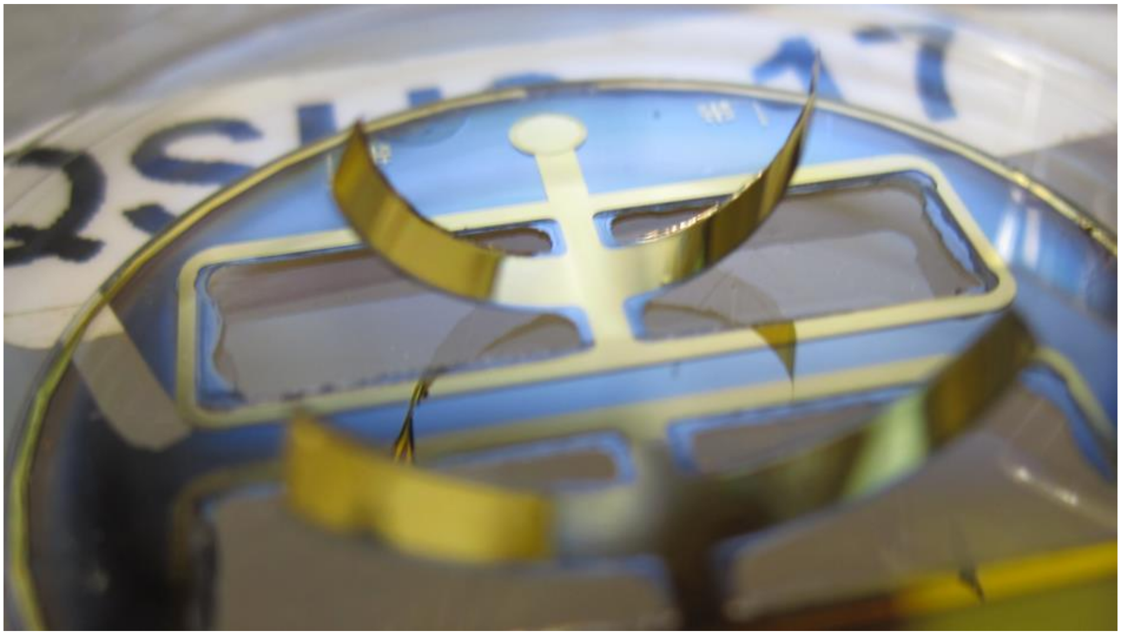


Figure 6-11: QSU8_17 released

The finished sample is then mounted on a PCB for testing, as shown in Figure 6-12. The back of the sample is electrically connected to the isolated central ring of the PCB via silver paste. Silver paste also connects very thin gold wires on the front contact pads for each wing pair to their SMA connector pads on the PCB. Because this board was designed with a central insulating pad which was slightly too small (both in its interior hole and its insulating circumference), black insulating paint was added around the edges of the central pad before the test structure wafer was affixed to the PCB to avoid shorting back of the wafer to the ground of the PCB. This isolated ground contact enabled three-terminal capacitance bridge measurements for each wing, but was later manually shorted to allow testing in the torsion bar test chamber, which only accommodated a single two-terminal contact.

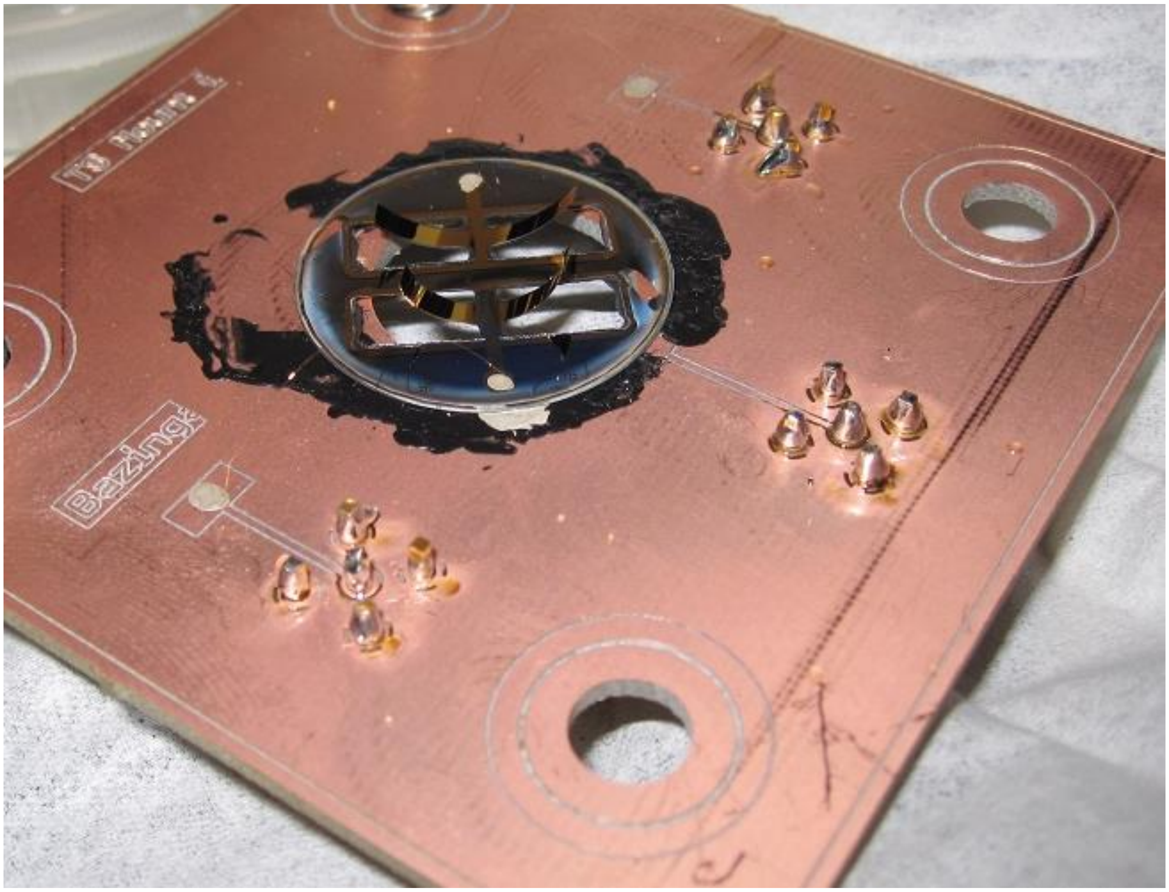


Figure 6-12: QSU8_17 mounted on its PCB for testing

Chapter 7 Experimental Measurement Procedure

The results of the optimization routine suggest that these resonant fluid diode wings *can* lead to successful flying microrobots. The true test of this model would involve actually *building* such a robot, but before doing that it makes sense to test the predictions that the analytical framework relies on, namely:

1. The wing's **bending angle** given its length, layer thicknesses, Young's moduli, Poisson ratio, and built-in strain,
2. The wing's **resonant frequency** given its length, layer thicknesses, Young's moduli, Poisson ratio, bending angle, and density,
3. The wing's **fluidic power dissipated given** its Reynolds number, bending angle, and amplitude,
4. The wing's **net force** output given its Reynolds number, bending angle, and amplitude, and
5. The wing's **BvD electrical parameters** given its length, layer thicknesses, fluidic power dissipated, and excitation frequency.

The completed wing pair QSU8_17a was electrically insulating and had a large static bending angle, so was chosen for testing. In order to test these predictions, QSU8_17a was electrically actuated by passing a sine wave from a function generator through a 1:20 amplifier at output voltage amplitudes of up to 200 V and frequencies of 10 to 200 Hz. The current flowing through these wings were measured and compared to that of a capacitor by using an SR-830 lock-in amplifier. The flapping wings' amplitudes were measured by directly filming them with a high-speed camera, and by observing the blur made by the wing envelop with a regular speed camera. The net force output by the wings was measured with a custom-designed torsion bar similar in concept to the one Cavendish used to measure the universal gravitational constant *G*, all enclosed in a vacuum chamber which allowed testing in N_2 at various levels of vacuum.

This chapter explains the measurements made on QSU8_17a and how they were taken.

Section 7.1 Measurements

Where a "test" is comparing a measured to a predicted result, a "measurement" is acquiring that measured result in the first place.

The wings' physical dimensions were measured, and several of its flight-related parameters were measured vs. applied frequency *f*, applied voltage *V*, and ambient gas pressure *P*:

1. Static bending angle and flapping amplitude
2. Electrical parameters
3. Output net force

This section will go through each of these quantities in turn, both explaining how they were measured and presenting the measured results.

Section 7.1.1 Physical dimension measurements

The length and width of the wings were measured by the mask dimensions. Specifically, referring to Figure 6-9, the wingspan is 15.68 mm, and the minimum bridge width is 1.48 mm, so that the effective wing length is 7.64 mm (the fillets have 0.5 mm radius, but do not affect the effective length or width for the purpose of resonant frequency or curvature). All of these are limited by the precision of the waterjet cutting machine and the RIE lateral etching for uncertainty of 0.1 mm.

The quartz and SU-8 thickness are determined by optical profilometry, while the Au & Cr layer thickness are determined by the quartz crystal microbalance in the metallizer at deposition time.

Thicknesses with uncertainty where SU-8: 7.6 +/- 0.12 μm and Quartz: 7.6 +/- 0.10 μm. In terms of the total thickness, the uncertainty is $\sigma_h = 0.22\mu\text{m}$. Total thickness is

15.55 µm, so relative uncertainty is 0.22/15.55 = 1.4%. Uncertainty on the metal layer thickness was significantly smaller due to their relatively smaller size, and does not affect the wing thickness uncertainty.

The wing length of 7.64 +/- 0.1 mm has a relative uncertainty of 1.3%. The uncertainty in the predicted resonant frequency due to uncertainty in the dimensions is $f_{res} \propto \frac{h}{L^2} \rightarrow \sigma_{fres} = \sigma_h + 2\sigma_L$ (in terms of relative uncertainties). Substituting the relative uncertainties in thickness and lengths give a predicted frequency relative uncertainty of 4%.

Section 7.1.2 Static Bending Angle and Amplitudes

Static bending angle was measured by taking pictures of the wing and measuring angles using ImageJ (software to measure angles between selected points in an image), as shown in Figure 7-1. By measuring the tip angle compared to the horizontal when the wing is open and closed, it is possible to calculate the wing's static angle and bending amplitude, knowing its length and the bridge width.

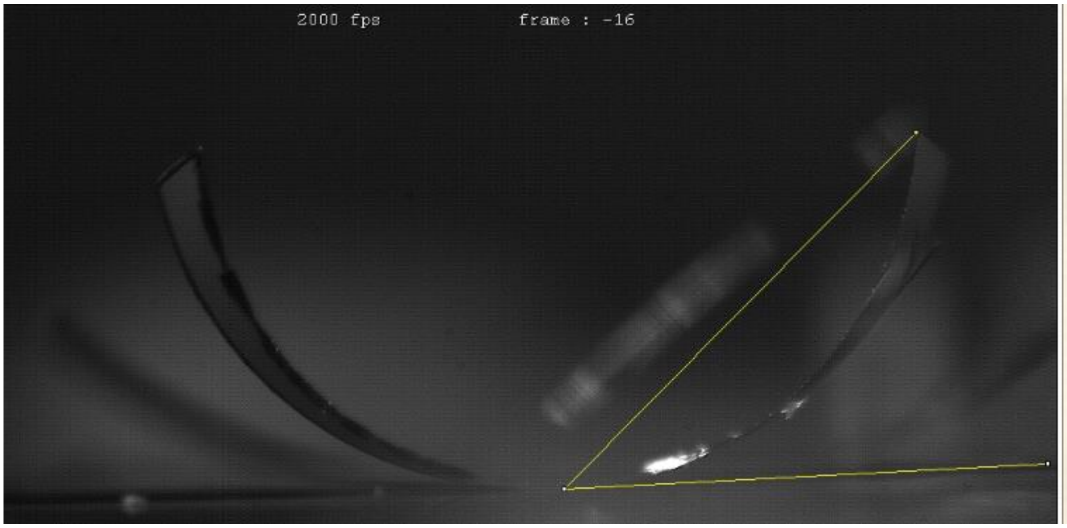


Figure 7-1: Measuring the wing angle using ImageJ

Comparing the amplitude produced for a given voltage at different frequencies is also a way to locate the wing's resonant frequency. Consider Figure 7-2, which plots the measured wing amplitude per applied volt of input signal vs. applied frequency for QSU8_17 in 1 atm N_2. This plot tells us the output wing amplitude per input voltage by looking at the low-frequency results, and identifies a resonance at 130 Hz.

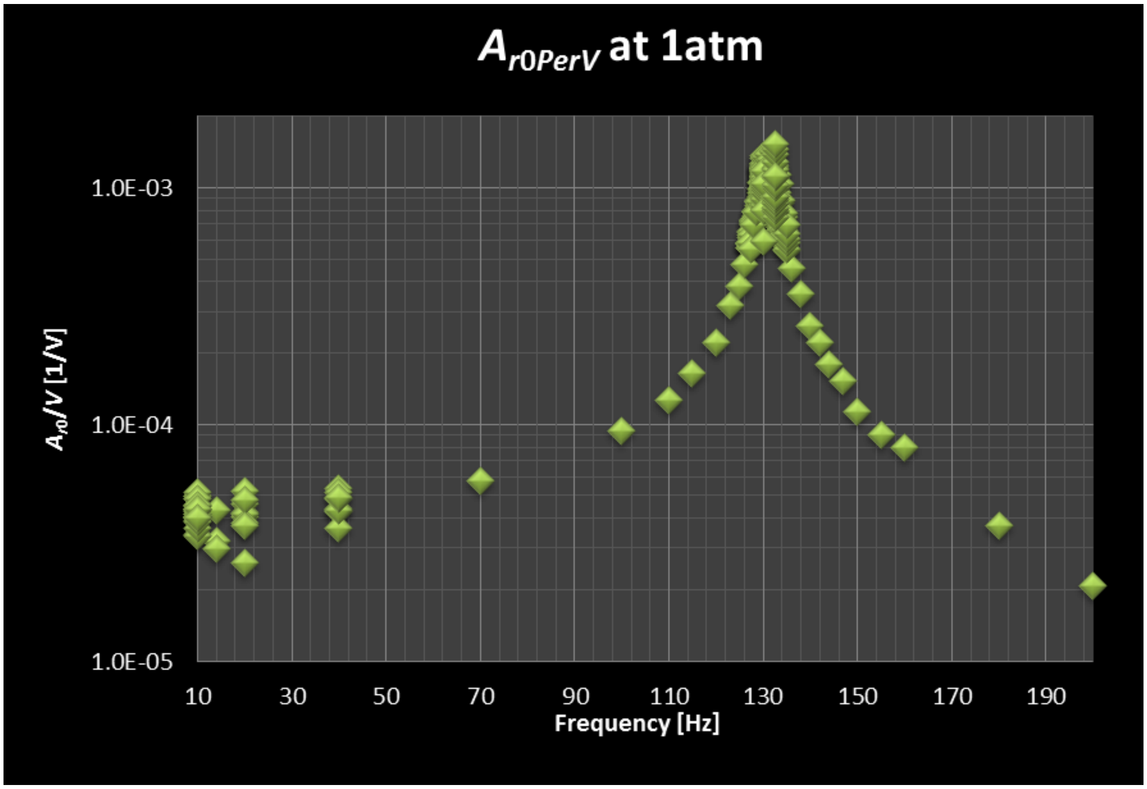


Figure 7-2: Measured wing amplitude per voltage shows a steady value at low frequency and an approach to resonance around 130 Hz

By dividing the output amplitude by the amplitude which would be produced at that voltage off-resonance, we arrive at the amplitude amplification quality factor, plotted in Figure 7-3. This plot illustrates several interesting things about the wings. It tells us the resonant frequency at each voltage, and the magnitude of the quality factor at that resonant frequency, as well as the quality factor's off-resonant behaviour around it. The resonant amplitude is sublinear in applied voltage, as expected for nonlinear damping;

this is shown by the quality factor being lower for the higher voltage sweeps. Also, the location of the peak shifts to the left for higher voltages, illustrating nonlinearity in the resonant frequency.

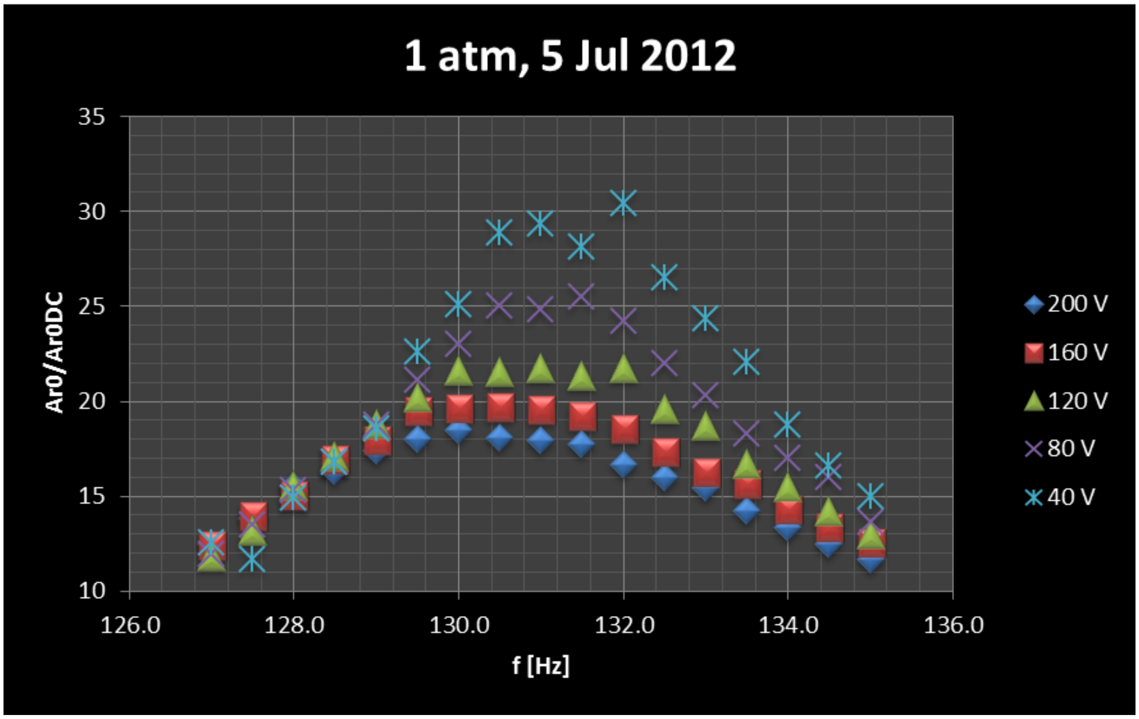


Figure 7-3: Amplitude amplification over DC amplitude (i.e., quality factor) vs. frequency at different applied voltages at 1 atm in N_2 demonstrates resonance, as well as its nonlinear behaviour

Section 7.1.3 Electrical Parameters Measures

To electrically test the wings, the current through the wings was measured and compared to the current through a decade capacitance box by differencing with an SR830 lock-in amplifier. Using an off-resonance input signal to the wings, the capacitance on the decade capacitance box was adjusted until it zeroed out the signal from the wings. This allowed more sensitive detection of the changes in current through the resonant branch around resonance, which are otherwise difficult to detect next to the current through the very large package capacitance.

The details of the setup and calculations for the electrical test procedure are explained in Appendix P. This leads to current measures through the resonance branch like those given in Figure 7-4.

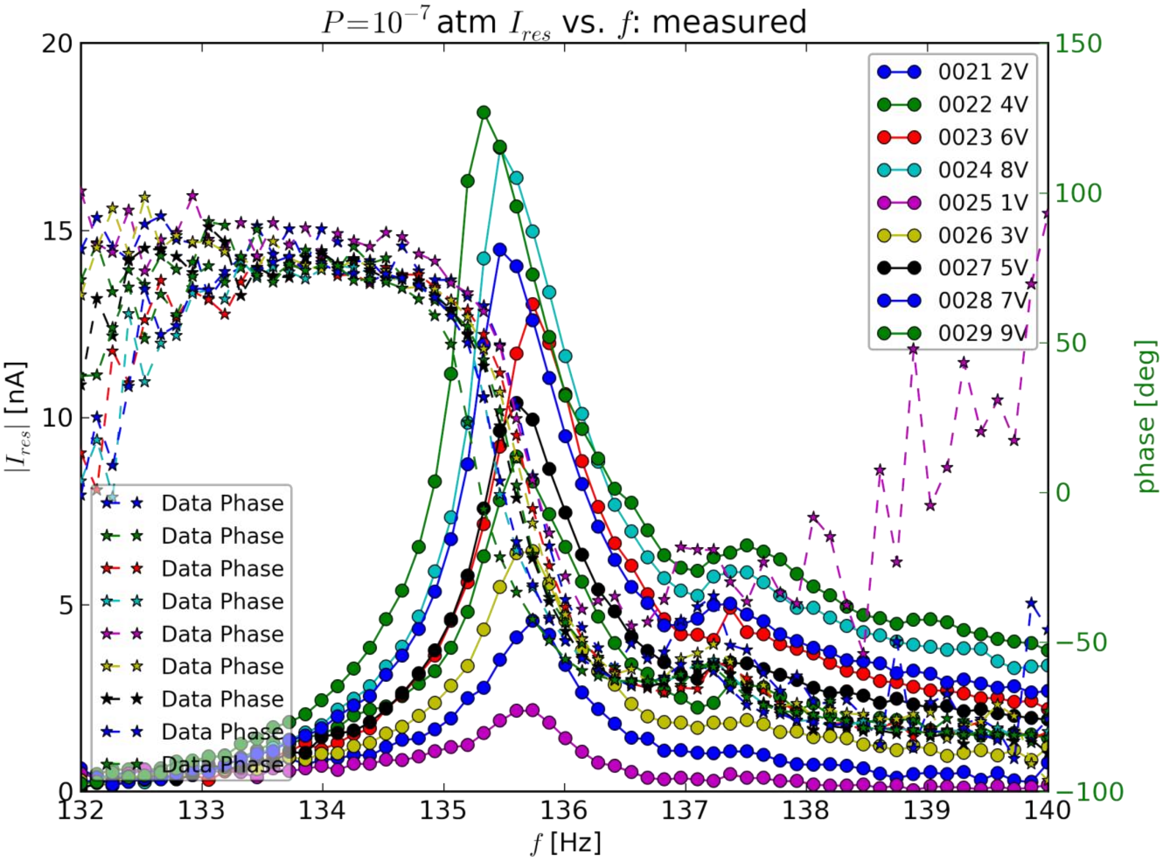


Figure 7-4: Resonance branch current for QSU8_17 at various low voltages in 1e-7 atm

Referring to the legend, each plot was at a different voltage (2, 4, 6, 8, then 1, 3, 5, 7, 9 V), and this was sweep numbers 21 - 29.

The resonant frequency appears independent of voltage until voltage rises above about 4V, at which point it starts to decrease with increasing voltage. Unfortunately, voltage is not the only determinant of resonant frequency. Notice that the 8V and 6V sweeps appear to be shifted to the right a bit compared to the 9V, 7V, and 5V sweeps.

This could indicate a decrease in resonant frequency with time; the odd-voltage sweeps numbered 25-29 were completed an average of two days after the even-voltage ones (25 & 26 vs. 23 & 24 of May 2016). This shift could also have been influenced by temperature. The 8 & 6V sweeps were at a lower temperatures than 5, 7, and 9V ones, as shown in Table 7-1. Temperature was measured continuously via a resistive temperature detector.

Table 7-1: Temperature vs. sweep number for current measures

Sweep#	Applied Voltage [V]:	Average Temperature [^{o}C]: ±0.2°C
29	9	23.30
24	8	22.87
28	7	23.36
23	6	22.42
27	5	23.05
26	3	23.04
22	4	22.84

At high voltages in vacuum, the current measures stopped being consistent, as shown in Figure 7-5. Notice the nonlinear effect of reduced resonant frequency with increased amplitude observed as voltage increased up to 25 V.

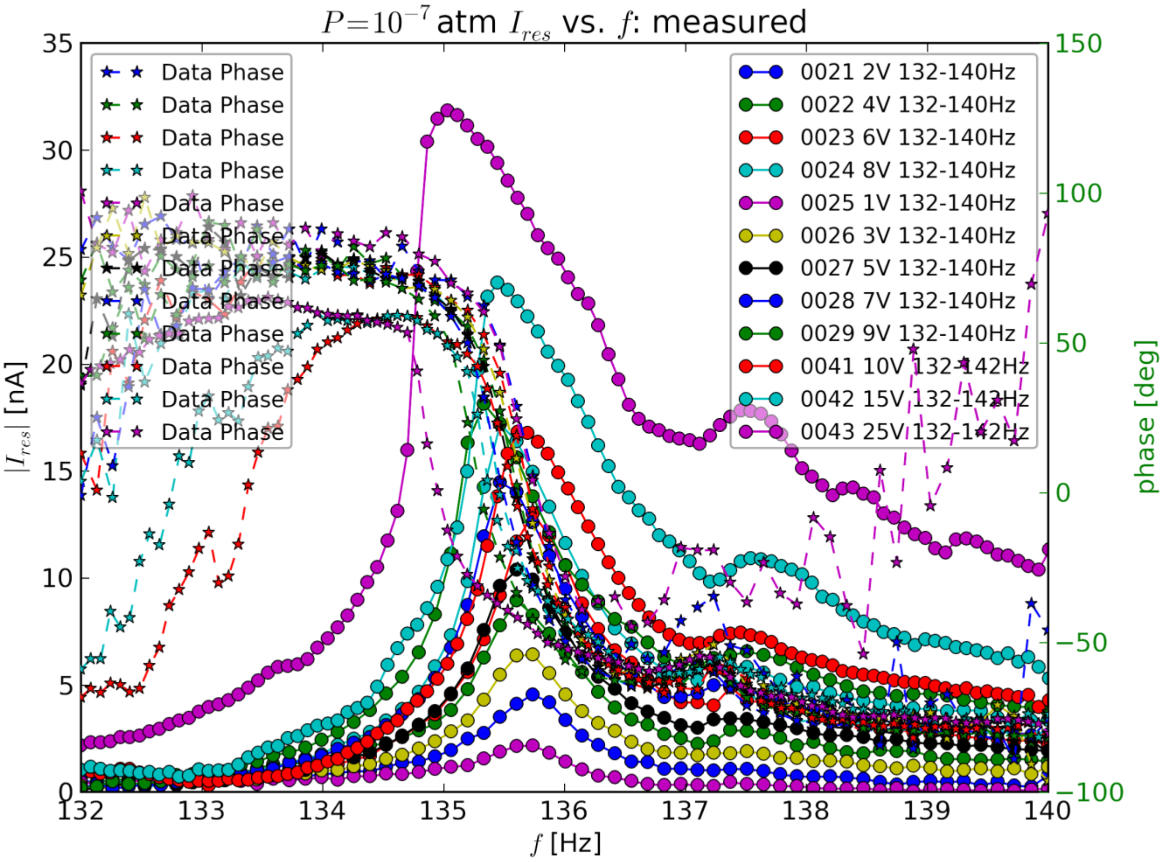


Figure 7-5: Electrical measurements of current through the resonant branch vs. frequency for different voltage amplitudes in vacuum showing highly nonlinear behaviour

Section 7.1.4 Force Measurement: The Torsion Bar

To measure the wing force, I placed the wing at one end of a light metal bar suspended from a wire and balanced with a counterweight. Since the wire has a low resistance to twisting, the force output by the wing will cause the bar to rotate dependent on the force and the wire's spring constant in torsion. By measuring the twisting of the bar using a laser reflected off of a mirror at its centre, we can determine how much net force the wing is producing. The details of the torsion bar theory and setup is explained in Appendix T.

The torsion bar was placed inside a sealed chamber to allow testing the wing's behaviour in various gas environments and ambient air pressures, and also to eliminate interference from moving room air. This chamber is shown in Figure 7-6. The spire extending up from the top of the chamber holds the torsion wire, which is a coaxial cable. This eliminates the need for separate wires to connect to the torsion bar, which is necessary because those wires can interfere with the very small forces being measured.

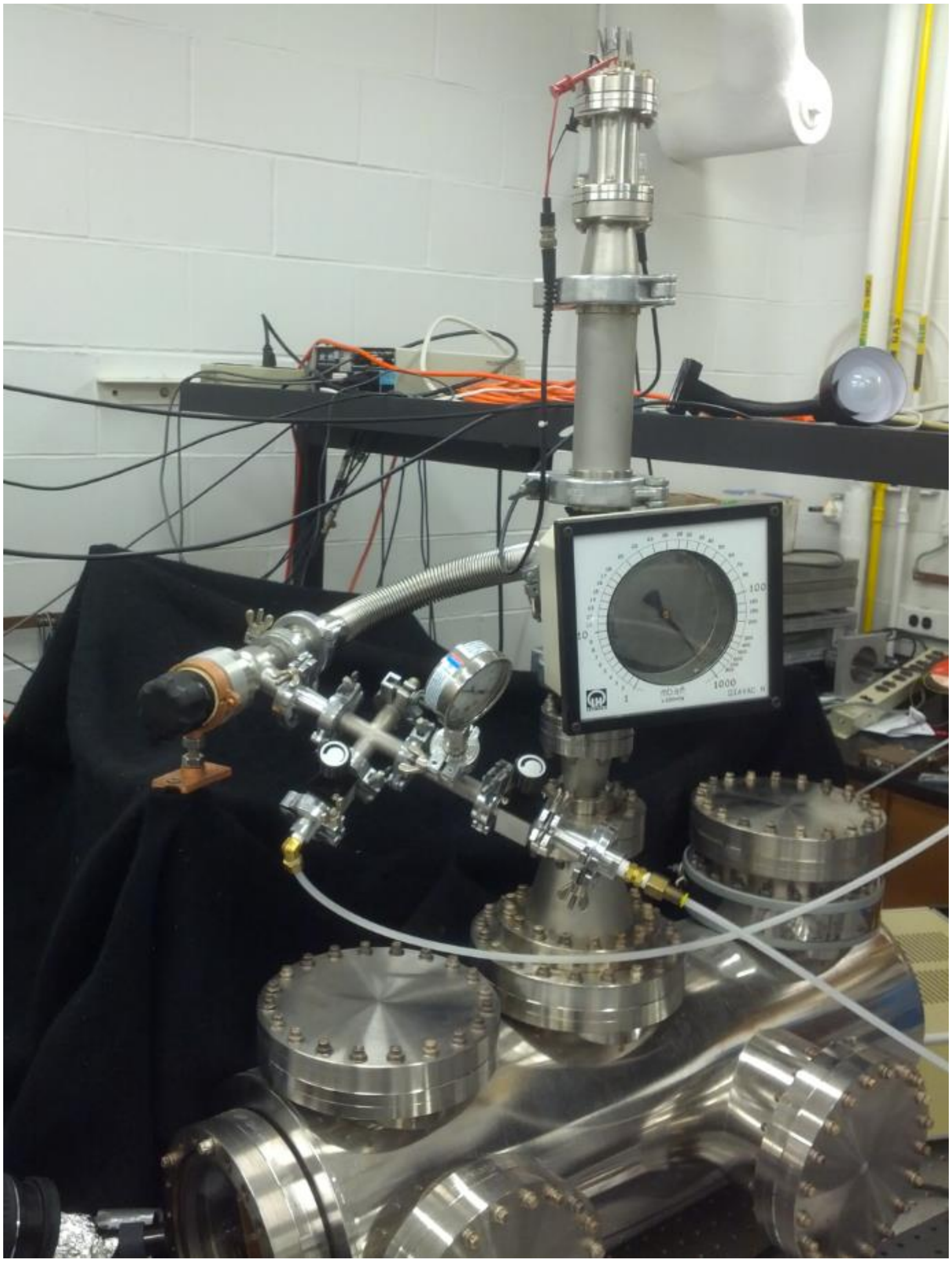


Figure 7-6: The vacuum chamber containing the torsion bar

The torsion bar appears in Figure 7-7, as viewed from the viewport where the camera photographed the wings in motion. This figure shows the sample arm (foreground), counterweight (background), and the junction block which holds the mirror, as well as the magnetic damper system below it. This contains sample QSU8_18 along with a paperclip holding down paper to hold down one pair of wings for ease of automatic image processing.

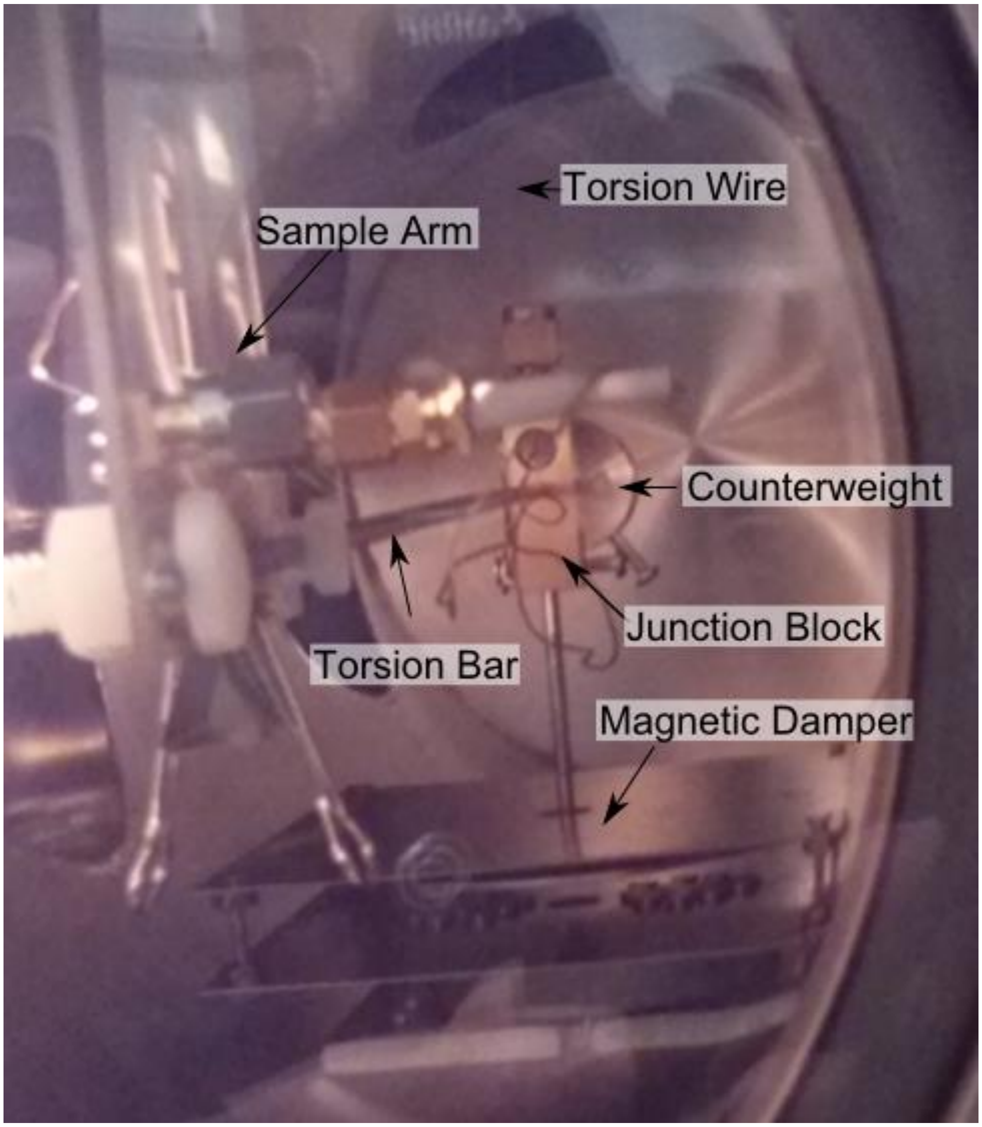


Figure 7-7: The final torsion bar

The rotation of the torsion bar was detected by the movement of a HeNe laser reflected off of a mirror placed on the junction block at its centre. The laser is directed to impact a position-sensitive detector, as shown in Figure 7-8. When the wings are actuated, the small forces they generate cause the torsion bar to rotate, which changes the position of the beam on the detector.

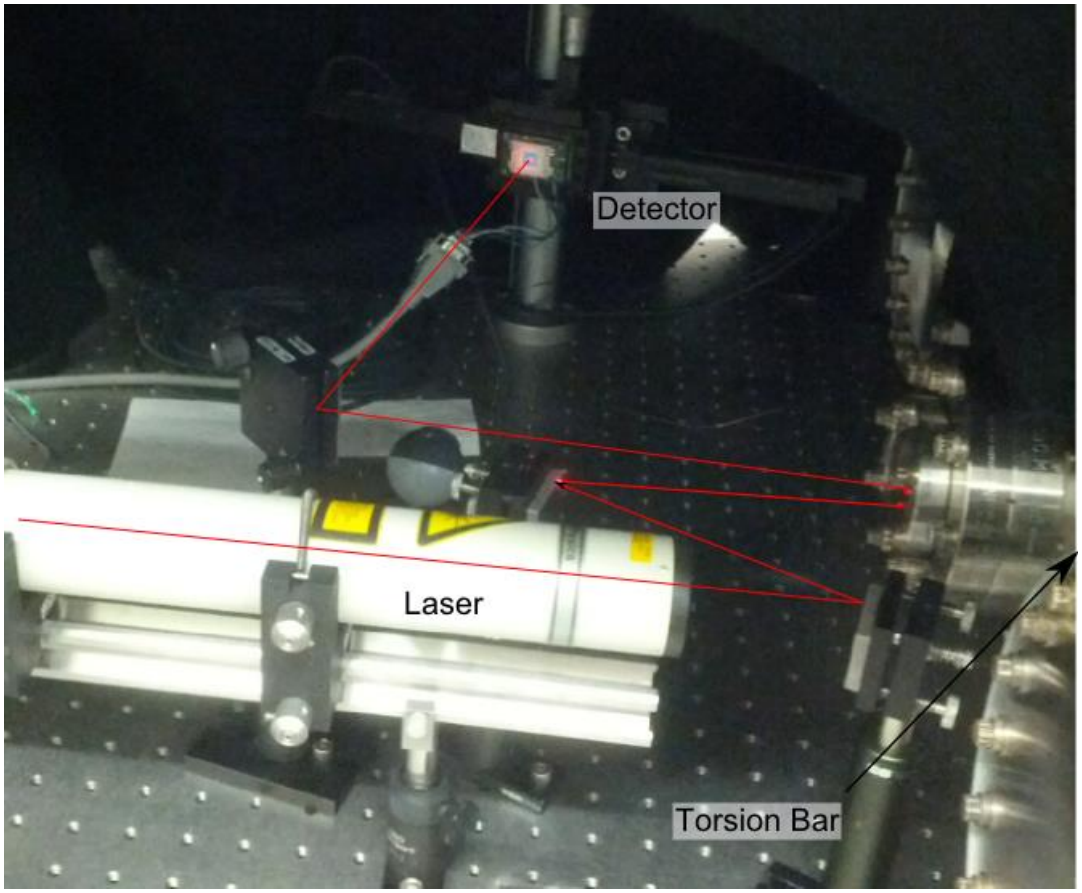


Figure 7-8: Laser position detection of the torsion bar's rotation

In order to improve the accuracy of the force measurement, the wings are modulated on and off. Varying the modulation period of the signal applied to the wings is a way to determine the frequency response of the torsion bar, as in Figure 7-9. From accurate measures of the torsion bar's component masses and their position, we can determine its moment of inertia. This, combined with its resonant frequency from its frequency response curve enables us to determine the effective spring constant of the torsion bar. Combining the torsion bar's spring constant with the known distances from the bar's centre to the photodetector allows turning the measured photodetector signal into a force measurement for the wings.

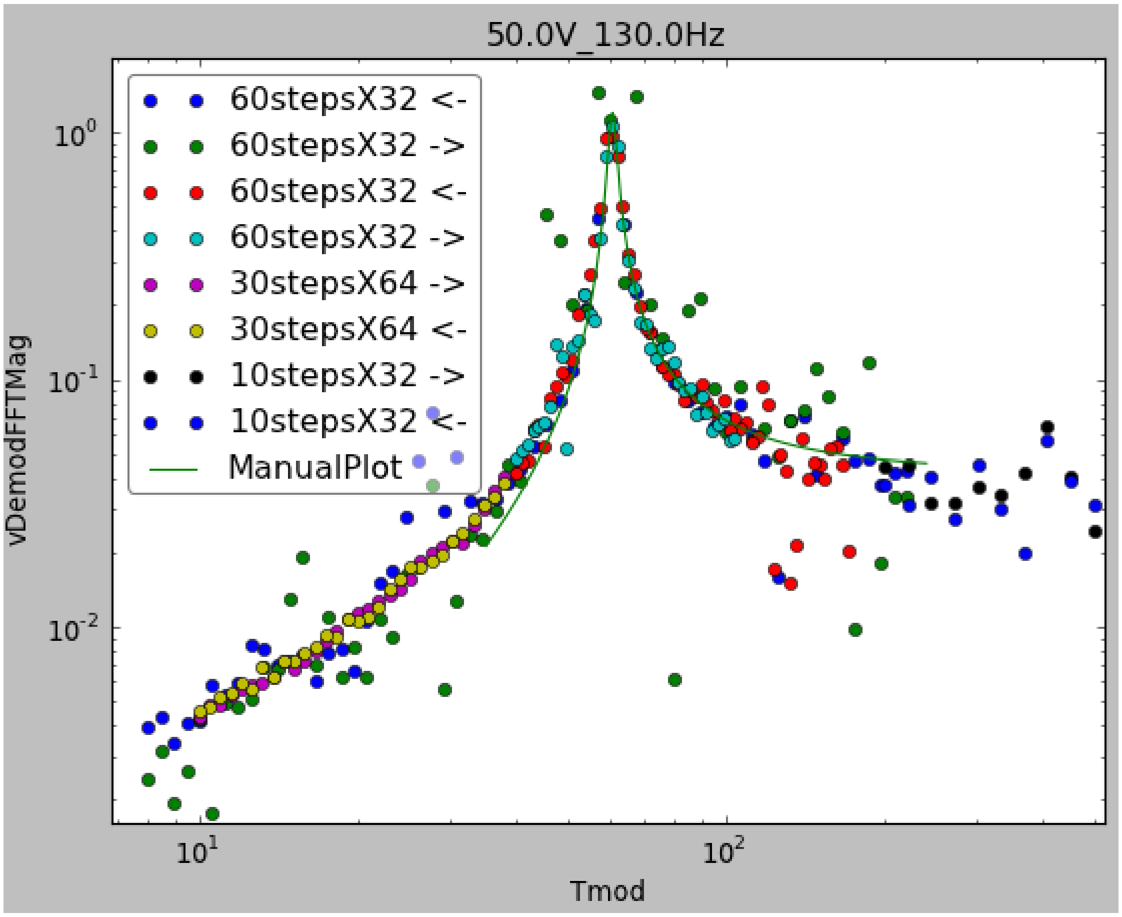


Figure 7-9: Torsion bar response voltage signal vs. wing modulation period at 50 V and 130 Hz

Using the calibrated torsion bar, I swept wing frequency and applied voltage to determine the output force, generating plots like Figure 7-10.

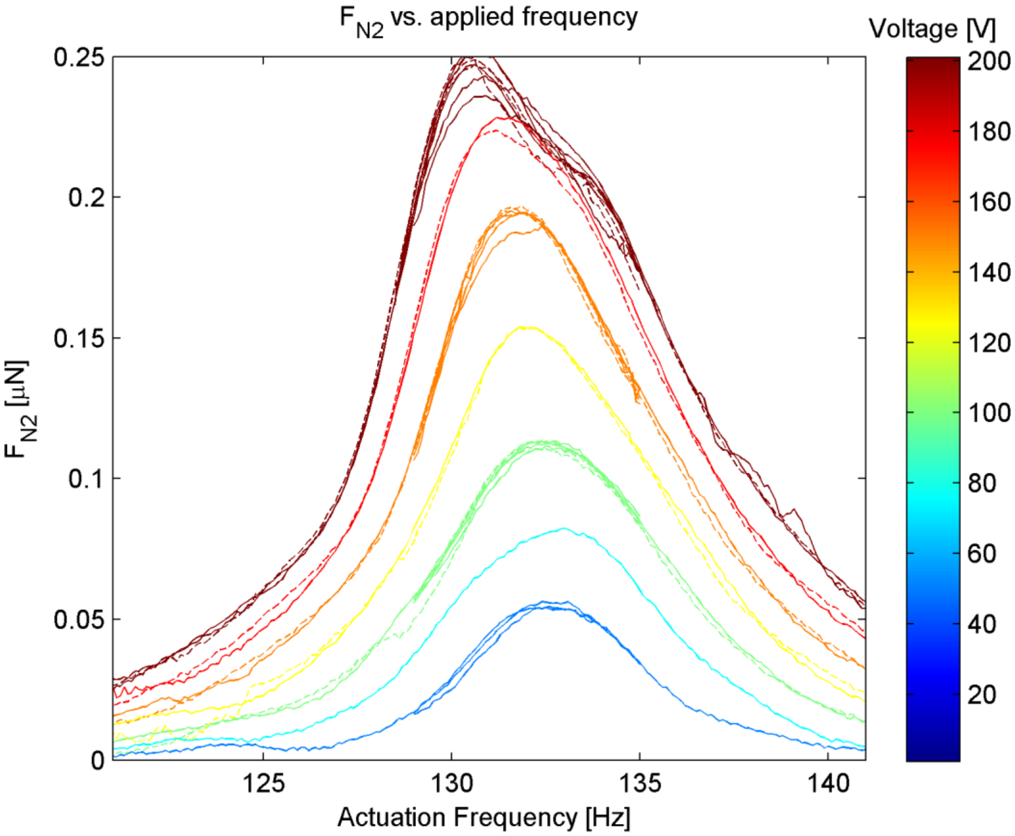


Figure 7-10: Measured wing output force in 1 atm N_2 vs. frequency for several up and down sweeps, from 50 V to 200 V in 25 V steps

Chapter 8 Experimental Results: Tests of Theory

This chapter analyzes the results in light of several tests of theory:

1. Natural bending angle ϕ_0
2. DC Voltage translation into extra curvature
3. Resonant frequency
4. Quality factor magnitude: Testing F_P by measuring the amplitude we *get* vs. the *expected one* for each voltage at resonance compared
5. Wing net force
6. Electrical parameters

Section 8.1 DC Voltage translation into extra curvature

Does the DC bending amplitude agree with theory? To determine this, I measured the bending angle of the wing in response to applied voltage, and found that although the *resonant* properties of the wings changed over time and with pressure, the *low* frequency response was significantly more stable.

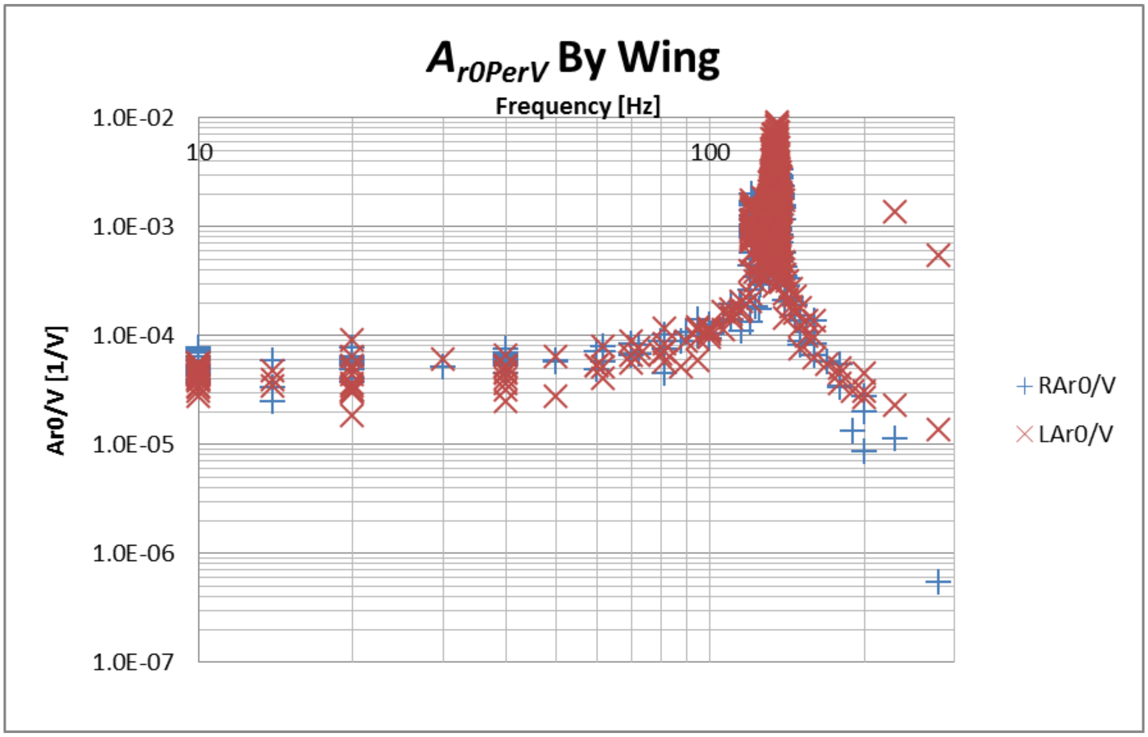


Figure 8-1: Quality factor by amplitude amplification, measured separately for each wing vs. frequency

At low frequency, the effective quality factor (i.e., amplitude amplification over DC) should be $Q_{DC} = 1$. Therefore, at low frequencies the measured flapping amplitude divided by the applied voltage tells us how voltages transduce into DC flapping amplitude:

$$A_{r0DCperV} \equiv \left.\frac{A_{r0}}{V}\right|_{\text{@low frequency}} = (4.8 \pm 0.7)\text{e-5 V}^{-1} \qquad (8.1)$$

where the uncertainty is one standard deviation of the measurements. The measures did not appear to be correlated with pressure or voltage.

This value can be used to calibrate the Young's modulus of SU-8, under the assumption that the multimorph bending angle theory is correct. Doing so yields a Young's modulus for SU-8 of 4.2 +/- 1GPa, (5.5e-5/V --> 5.2 GPa, 4.1e-5/V --> 3.4 GPa) which is well within the accepted range of 2.5 to 4.95 GPa reported in Section N.2. This large range in the uncertainty of SU-8 would have allowed amplitude amplification factors of 3.4e-5/V and 5.34e-5/V.

Section 8.2 Initial Bending Angle, ϕ_0

The wing's angle measures are plotted in Figure 8-2.

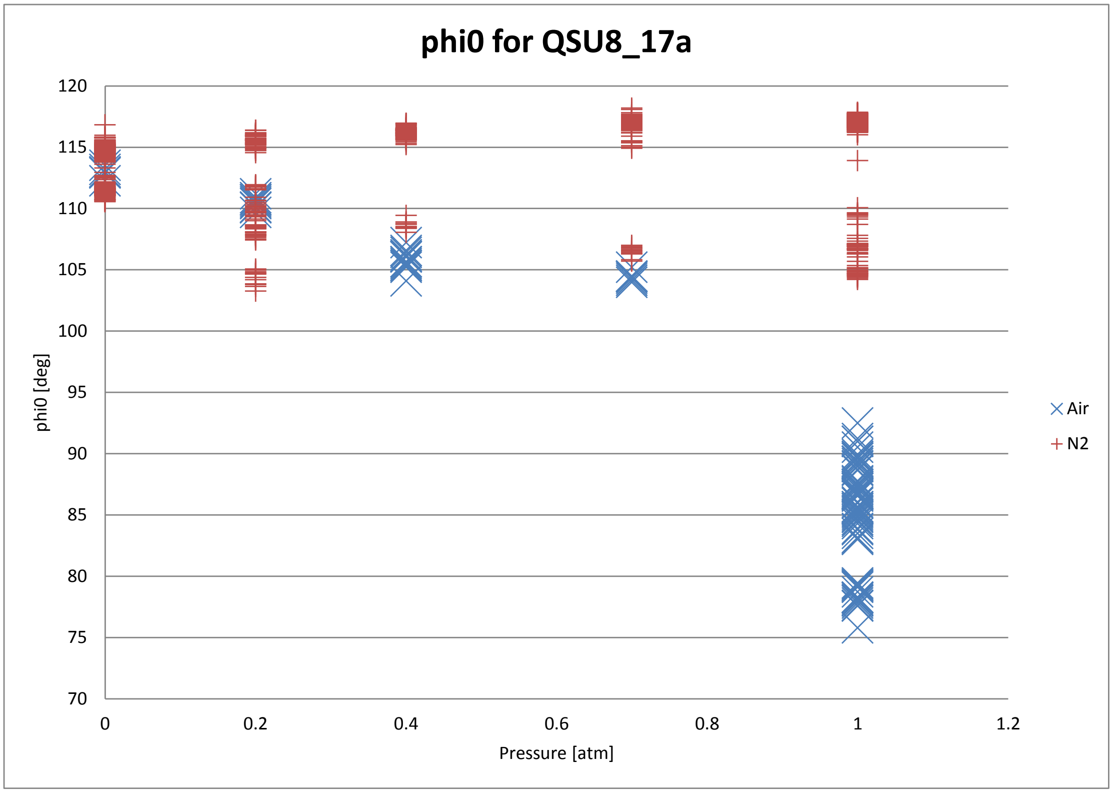


Figure 8-2: Static bending angle of QSU8_17a vs. measurement pressure in air and N_2 environments

The static bending angle of the QSU8_17a wings was not constant over various pressure and temperature conditions. Notably, the angle increased from as low as 76^o to as high as 117^o on pumping to partial vacuums, and did not completely (or sometimes at all) return to the lower bending angle on backfilling with N_2. The lower angle did return (to within experimental precision) on returning the wing to an ambient air environment. One possible explanation is that approaching vacuum causes water vapour to exit the SU-8, leaving it under increased tension, which the wing relaxes by increasing its curvature.

This is consistent with the wing maintaining this higher bending angle after backfilling with N_2 back to 1 atm.

The bending angle of the wing is largely dependent on the latent strain in the SU-8 after fabrication. The SU-8 strain reported in literature was, from Section N.3, 4.6e-3 to 11.1e-3. These lead to a large range of allowable initial bending angles (using the N_2-calibrated E_Y) of 55^o to 179^o. Using the average bending angle of 111^o to calibrate the SU-8 built-in strain yields **7.5e-3**: well within the accepted literature range.

Section 8.3 Resonant frequency

Using the calibrated SU-8 Young's modulus, the resonant frequency can be calculated using $f = \frac{\lambda^2}{2\pi}\sqrt{\frac{EI}{\rho_m Bh^3}}\frac{h}{L^2} f_{resCor}$. Doing so yields f_{res} = **132.6 Hz**, while the measured value ranged from 130 Hz to 136 Hz (in N_2) depending on voltage and pressure conditions, dropping with higher voltage and higher pressure.

The resonant frequency measurement was different for each wing, and was also affected by pressure and strongly affected by whether the wing was in room air or pure N_2. The right wing's resonant frequency was 1.2-1.8% higher than the left wing's, and had a smaller resonant amplitude. Examining high-speed wing videos shows that at both resonances, the right wing leads the left wing in phase by between 45 and 90^o. Since both resonances have a similar relative phase for each wing, this suggests that the two resonances are indeed separate resonances for each wing rather than an in-phase and out-of-phase resonant mode.

Section 8.4 Quality Factor Magnitude

By plotting the measured amplitude at resonant frequencies vs. the DC-amplitude at the same input voltage, we obtain the quality factor, as shown in Figure 8-3 (For details of how to make the calculations for these plots, see Appendix O).

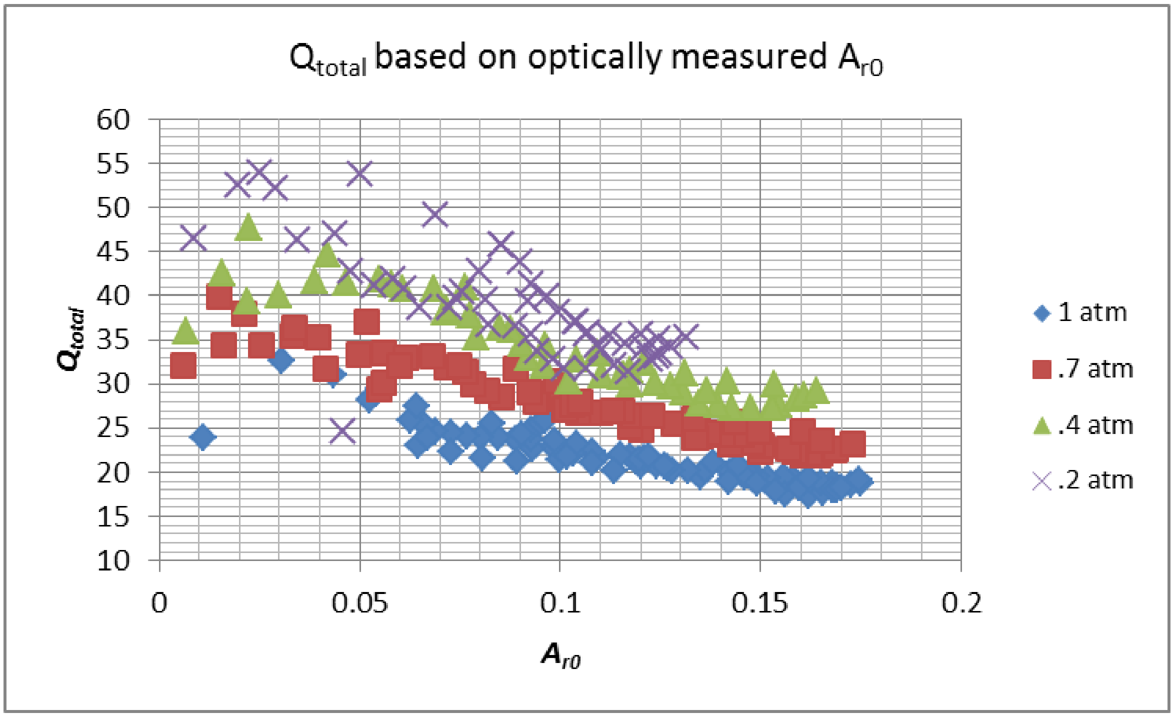


Figure 8-3: Measured quality factor by amplitude-amplification at resonance

All data for these plots is in N_2 by averaging the amplitude of both wings. Low amplitudes are difficult to measure optically. Further, because the resonant frequency depends on the pressure and amplitude, and because it changed over time and with temperature, not all data points are exactly on-resonance. The quality factor declined with amplitude at each pressure, and was lower for higher air pressures, as expected for nonlinear damping.

To quantify how well this agreed with the damping predicted by the CFX simulations, we need to calculate the what the quality factor from fluidic damping should have been for each amplitude and pressure. As calculated in Chapter 4:

$$Q_{fluidic} = \frac{2\pi K_{\max}}{TP_{avg}} = \frac{2\pi \frac{1}{2} L v_{rms}^2 \mu_m}{\frac{1}{f} \frac{1}{2} \rho_{air} C_{DP} A_{wing} v_{rms}^3}$$

$$Q_{fluidic} = \frac{L(\rho_m Bh)}{\rho_{air} C_{DP} (BL)} \frac{2\pi f}{v_{rms}} = \frac{\rho_m h}{\rho_{air} C_{DP}} \frac{\omega}{A_{r0} \omega L / (2 f_{resCor})}$$

$$Q_{fluidic} = \frac{2 f_{resCor} \rho_m h}{\rho_{air} C_{DP} A_{r0} L} \quad (8.2)$$

Incorporating the CFX-determined drag coefficients from Chapter 3, this formula generates the predicted fluidic quality factors shown in Figure 8-4.

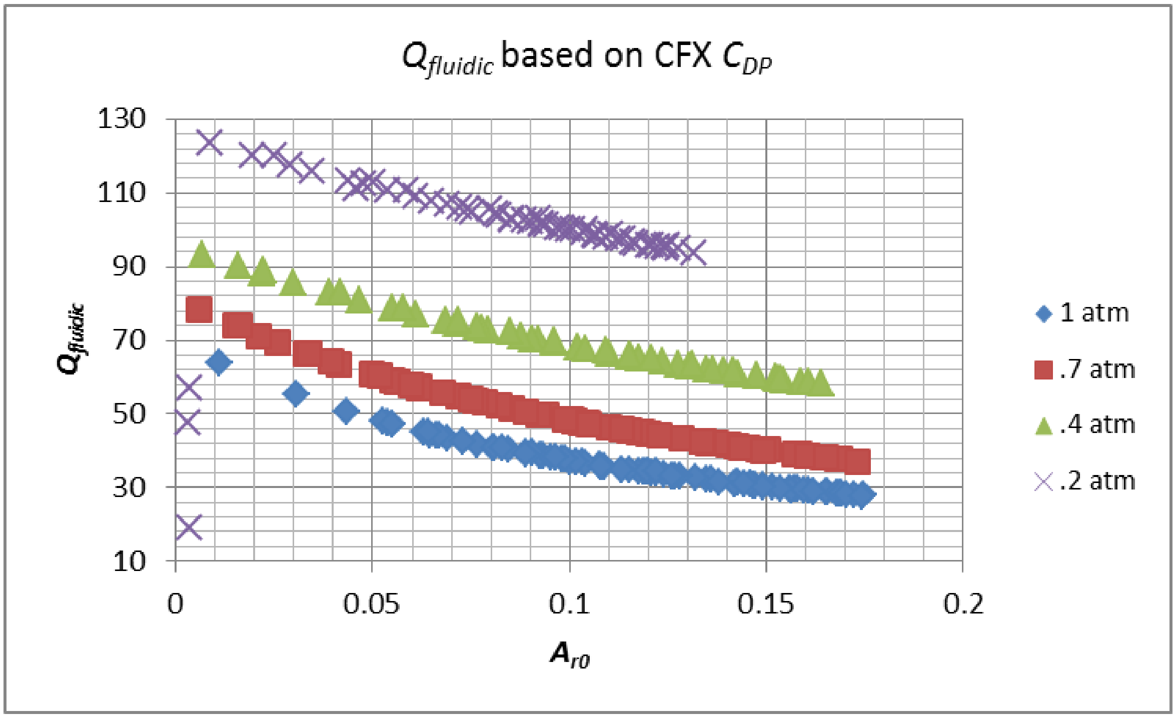


Figure 8-4: Expected quality factor at each measured amplitude and pressure based on fluidic damping

But the predicted quality factors in Figure 8-4 look nothing like the measured quality factors from Figure 8-3. What went wrong?

The analytical framework assumed mechanical losses were negligible next to fluidic ones. For nonnegligible mechanical damping, the effective quality factor is due to two sources of power dissipation. Writing down and manipulating the energy-balance quality factor definition reveals that we can combine quality factors like resistors in parallel:

$$Q = \frac{2\pi U_{\max}}{T\left(P_{fluidic} + P_{mech}\right)} = \frac{1}{\frac{TP_{fluidic}}{2\pi U_{\max}} + \frac{TP_{mech}}{2\pi U_{\max}}} = \frac{1}{\frac{1}{Q_{fluidic}} + \frac{1}{Q_{mech}}}$$

For the amplitude measures taken at resonance, Figure 8-4's predicted $Q_{fluidic}$ of 30 to 130 are dominant next to a typical air-damped MEMS cantilever Q_{mech} in excess of 7500 (e.g., [62]). However, the quartz-SU8 wings have some problems preventing high quality factors: SU-8 is not a very good resonator material ([63] found that the maximum mechanical quality factor of SU-8 cantilevers is around 70), and the two wings on QSU8_17a had similar but unequal quality factors due to imperfect mask alignment, leading to mode interference. These effects led to a non-ignorable mechanical quality factor for the wings. Unfortunately, eliminating fluidic damping for the wings in order to precisely measure Q_{mech} without any fluidic damping required pressures below 10 μBarr. After a number of system improvements in early 2013, I did attain this pressure in the test chamber, but the wings underwent permanent changes as a result, and on returning to 1 atm of air ambient conditions the wing flattened out completely, as shown in Figure 8-5. Even still, the amplitude at resonance did not increase significantly at these pressures, indicating that the mechanical quality factor for this sample is nonignorable.

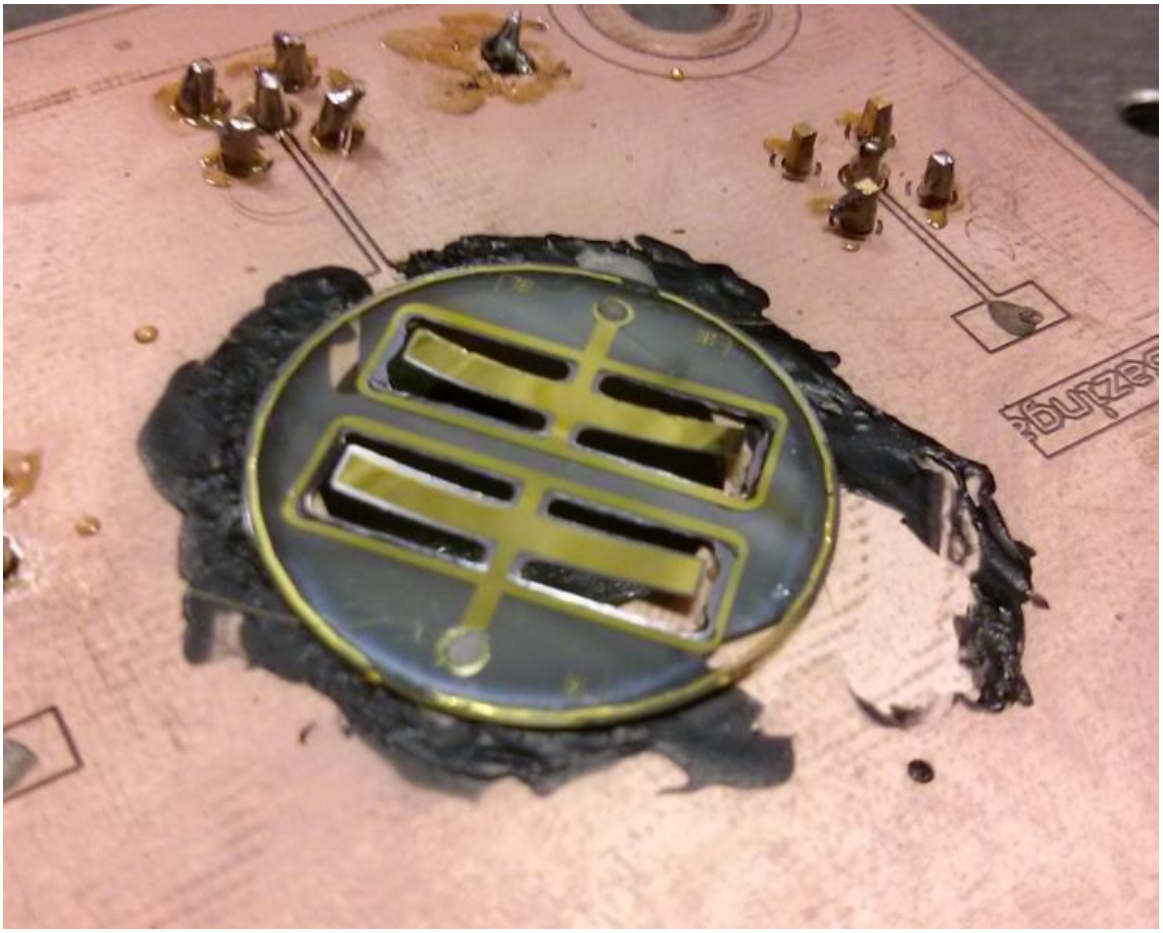


Figure 8-5: After almost continuous testing for 2 years and tirelessly flapping over 5 billion times, QSU8_17 settles into retirement in June 2013 after months at vacuum pressures lower than 1e-8 atm were too much for its built-in curvature

The changes in the wing with changing air pressure mean that the mechanical quality factor is not only non-ignorable , but also non-measurable unless the fluidic quality factor is known. One approach then is to use the CFX-predicted fluidic damping coefficient to calculate the fluidic quality factor for each amplitude measurement, then use this to calculate the what the mechanical quality factor must have been if the fluidic damping coefficient is correct. Figure 8-6 shows the result.

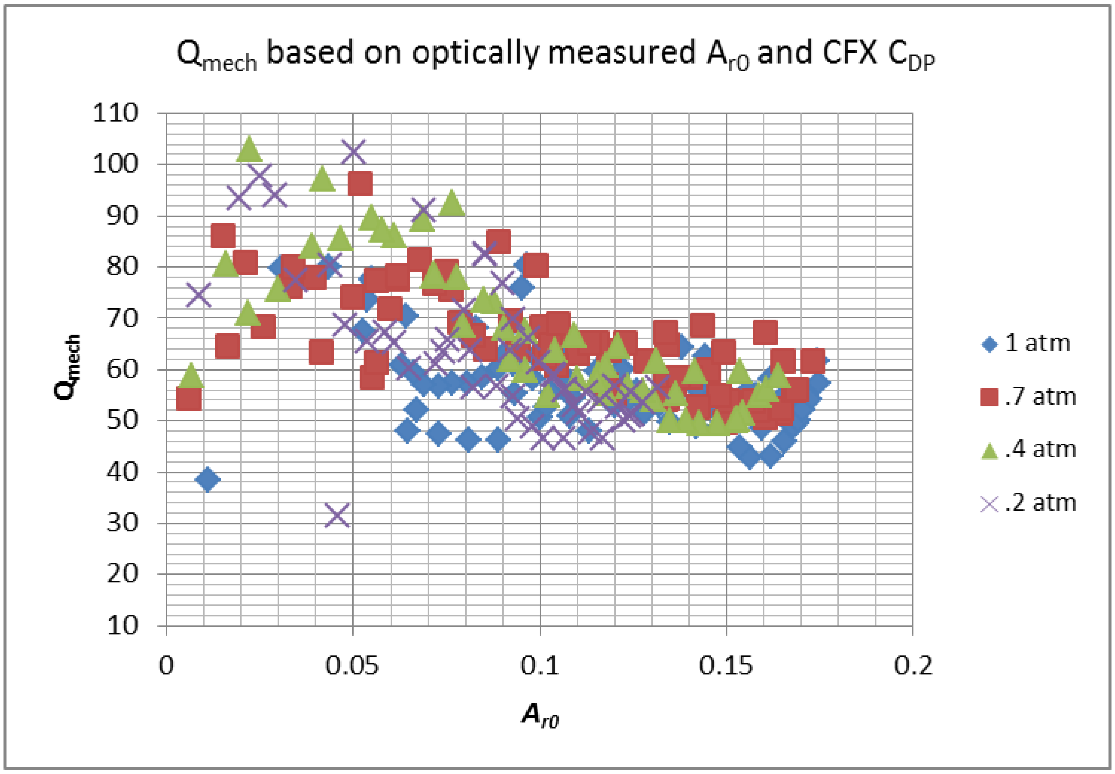


Figure 8-6: Quality factor due to mechanical damping

Compared to the Q_{total} plot, accounting for fluidic quality factor as-predicted has compressed the data from different pressures together. This is consistent with the mechanical damping-limiting hypothesis, since air pressure should have less effect on mechanical damping than it does on fluidic damping (unfortunately since this wing's properties change with air pressure we cannot say that air pressure should have *no* effect on mechanical damping). Unfortunately, this calculated mechanical quality factor is still not completely constant across all bending amplitudes and pressures. This could reflect a nonlinear mechanical damping at these amplitudes, but may also be due to the changed overlap of the two wings' resonant frequencies with amplitude and pressure.

Figure 8-6 suggests an average mechanical quality factor of 45-70 at moderate amplitudes. To compare the theory and measurement, we can use a fixed mechanical quality factor of 58, the $A_{r0DCperV}$ from Section 8.1 and measured amplitudes to calculate the power drag coefficient by rearranging (8.2) to get $C_{DP} = \dfrac{2 f_{resCor} \rho_m h / L}{\rho_{air} A_{r0} Q_{fluid}}$. Figure 8-7 plots the result.

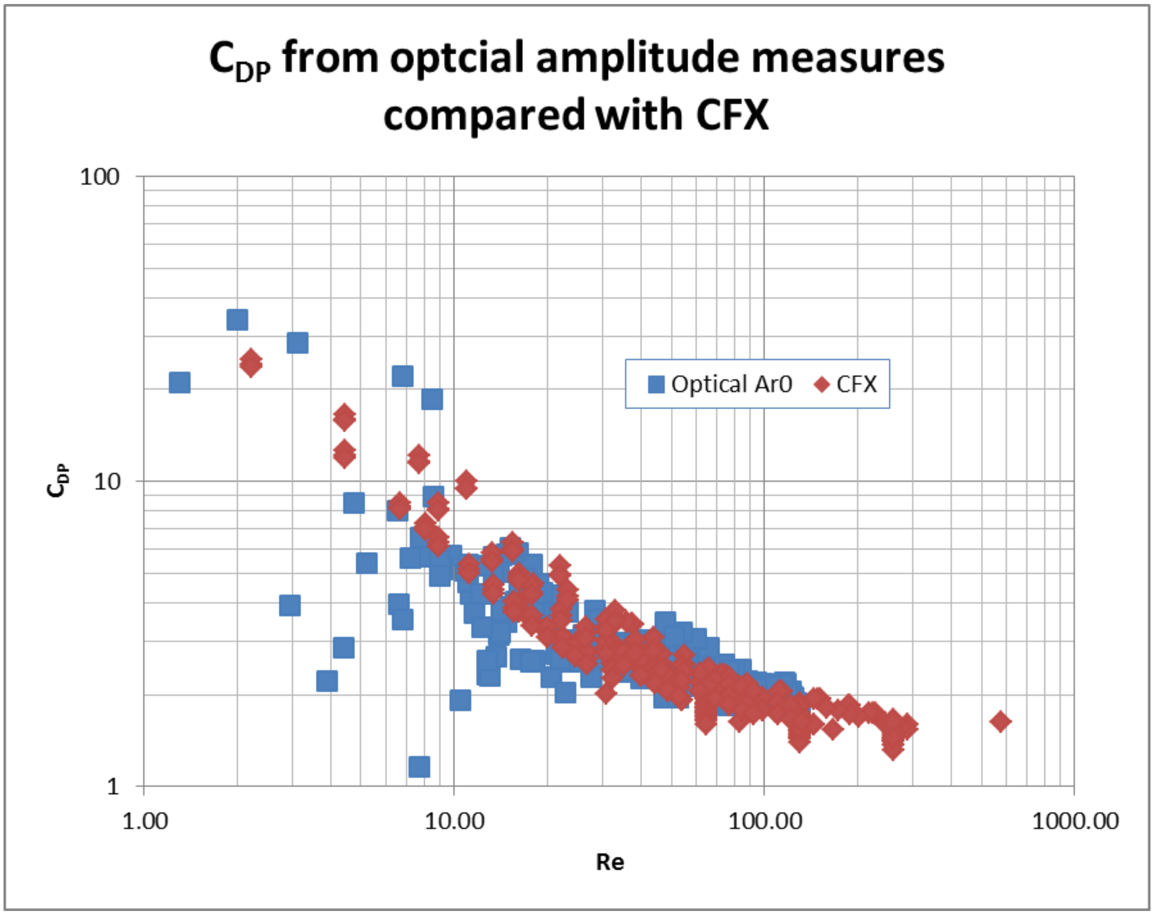


Figure 8-7: Drag coefficient from CFX vs. drag coefficient measured from optically-measured amplitudes and a fixed mechanical quality factor

Figure 8-7 shows that the calculated C_{DP} is consistent with the measurements. Further testing required reducing the uncertainty introduced by the SU-8 and the wing resonant-frequency overlap of QSU8_17a. Future work should improve on this by fabricating a cantilever out of materials whose properties are independent of ambient pressure to at least 1 μBarr, which would allow measuring the mechanical damping directly.

Section 8.5 Wing net force:

Using the torsion bar as described in Section 7.1.4, I was able to find the force produced by the wings under a variety of voltage, frequency, and pressure conditions. This test structure was mounted on a PCB which was partially blocking even the area directly behind one of the wings, as shown in Figure 8-8.

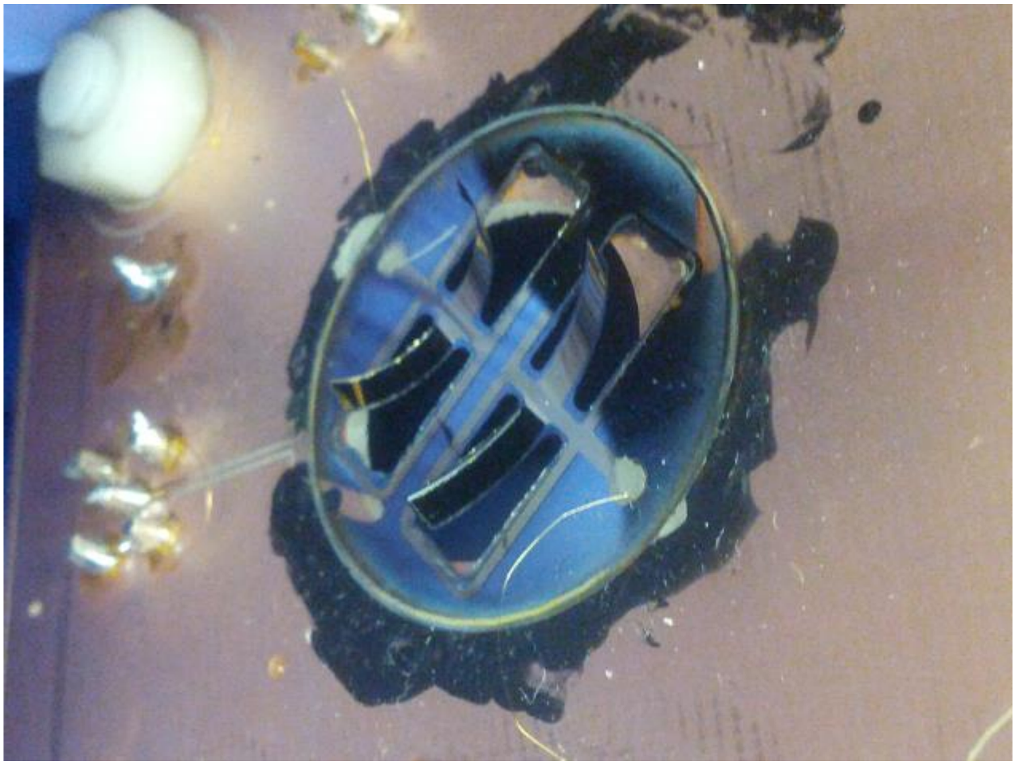


Figure 8-8: Tested wings were mounted with a significant amount of PCB blocking the airflow from the wings. QSU8_17a is in the back, and the right wing is particularly blocked by the PCB.

Technically, the torsion bar does not directly measure the force output by the wings, but rather the net force on the wings and the PCB together. When the wings flap, on-average they cause airflow to make a net force, but this also causes some air to hit the PCB, leading to a force in the opposite direction. Simulations which include the PCB reveal that this force reduction scales from 50-90%, with a complicated dependence on amplitude, frequency, and pressure. For consistency of comparing theory with measurement for the force data, plots will assume a uniform measured-force reduction due to the torsion bar of 75% for all cases (i.e., they assume that the actual wing net force is 1/75% = 4 times the reported force from the torsion bar, because the wings' air movement caused 75% of the force on the wings to appear on the PCB in the opposite direction).

Section 8.5.1 Measured vs. predicted force

Measured vs. predicted force at several related initial bending angles are plotted for different pressures of N_2 in Figure 8-9 through Figure 8-12.

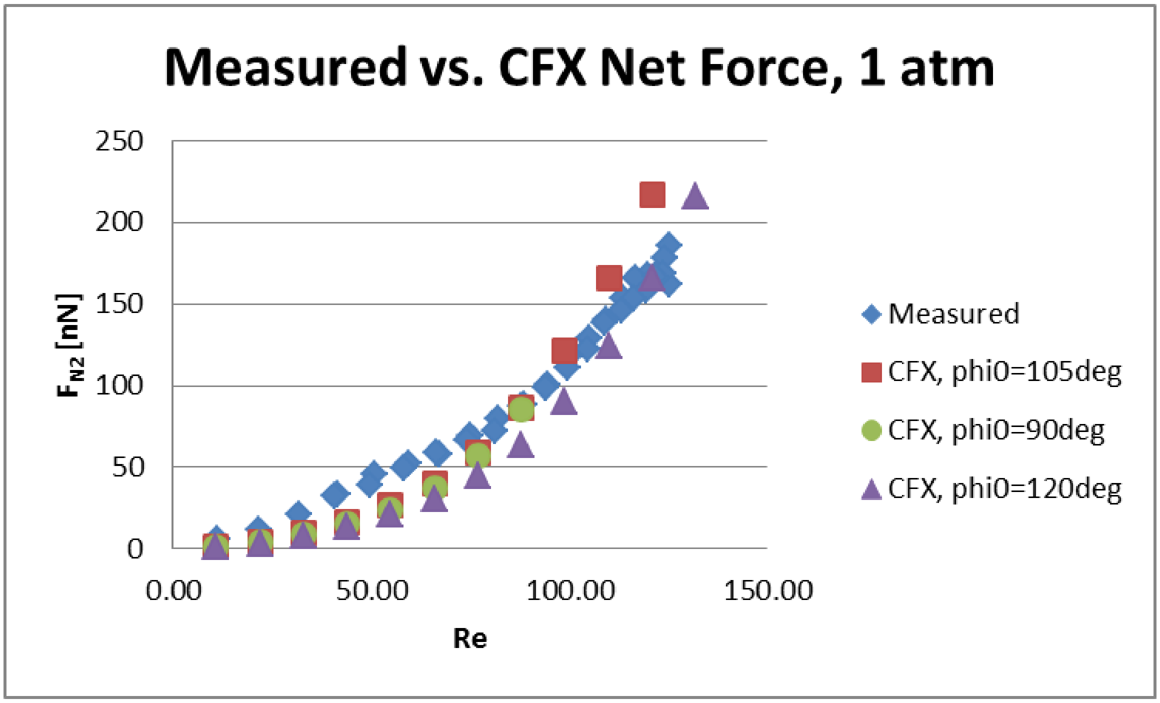


Figure 8-9: 1 atm N_2 measured vs. predicted force

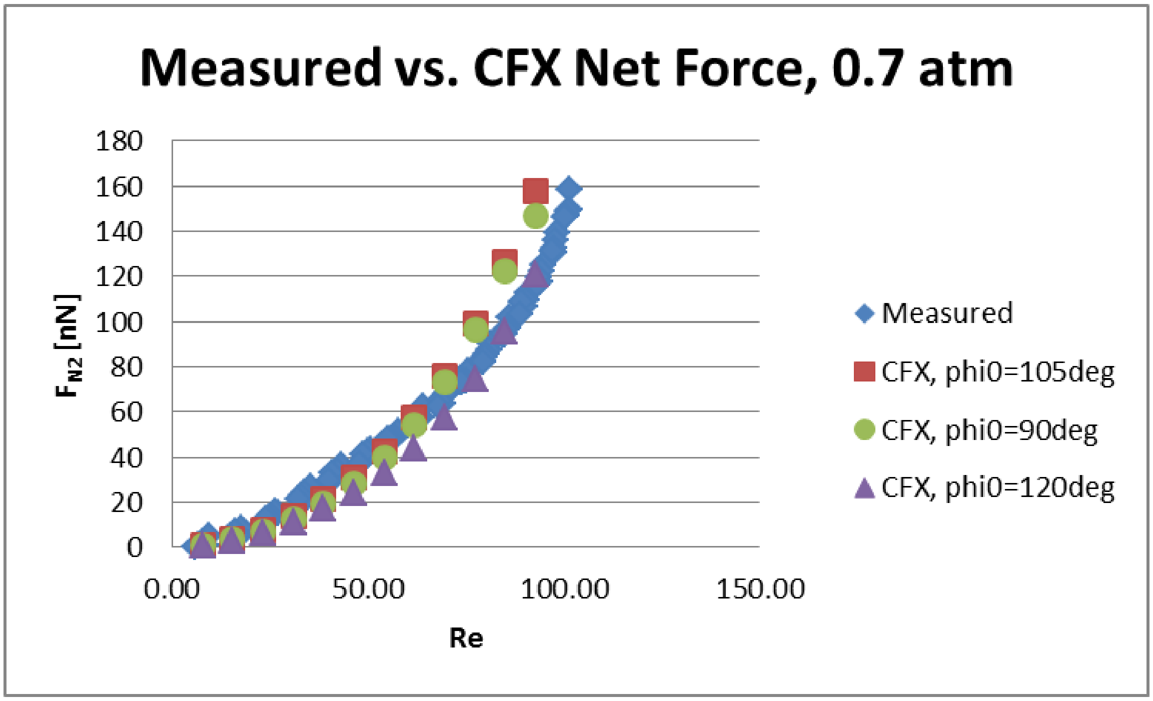


Figure 8-10: 0.7 atm N_2 measured vs. predicted force

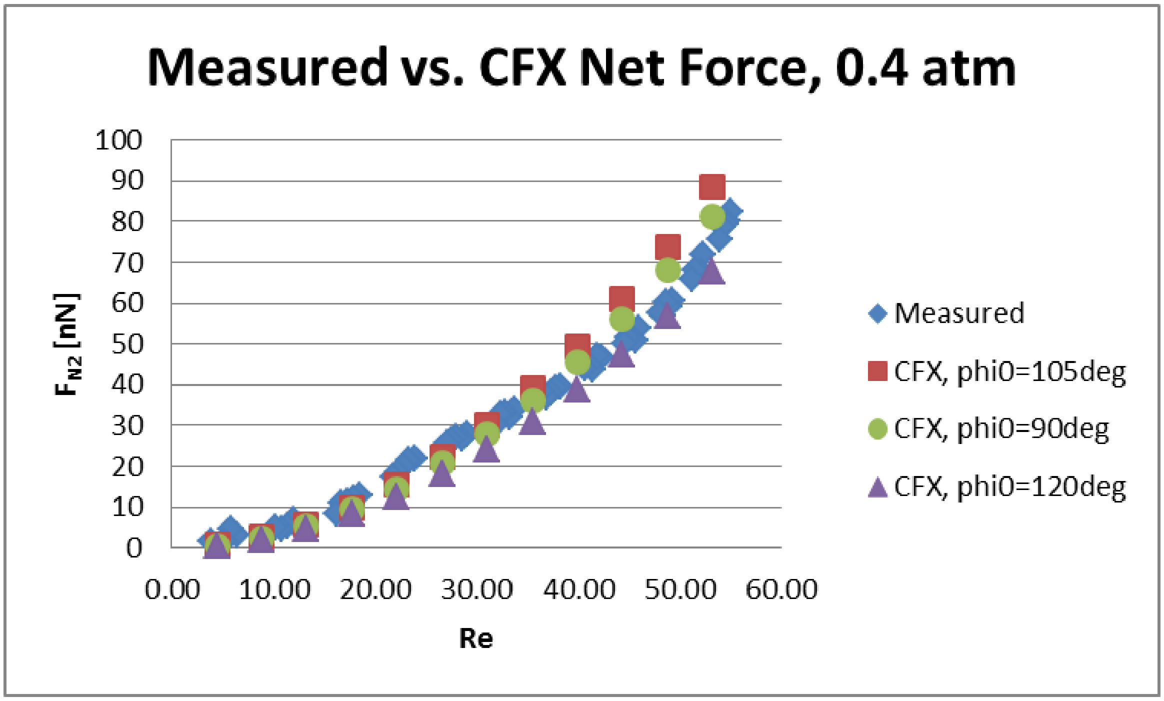


Figure 8-11: 0.4 atm N_2 measured vs. predicted force

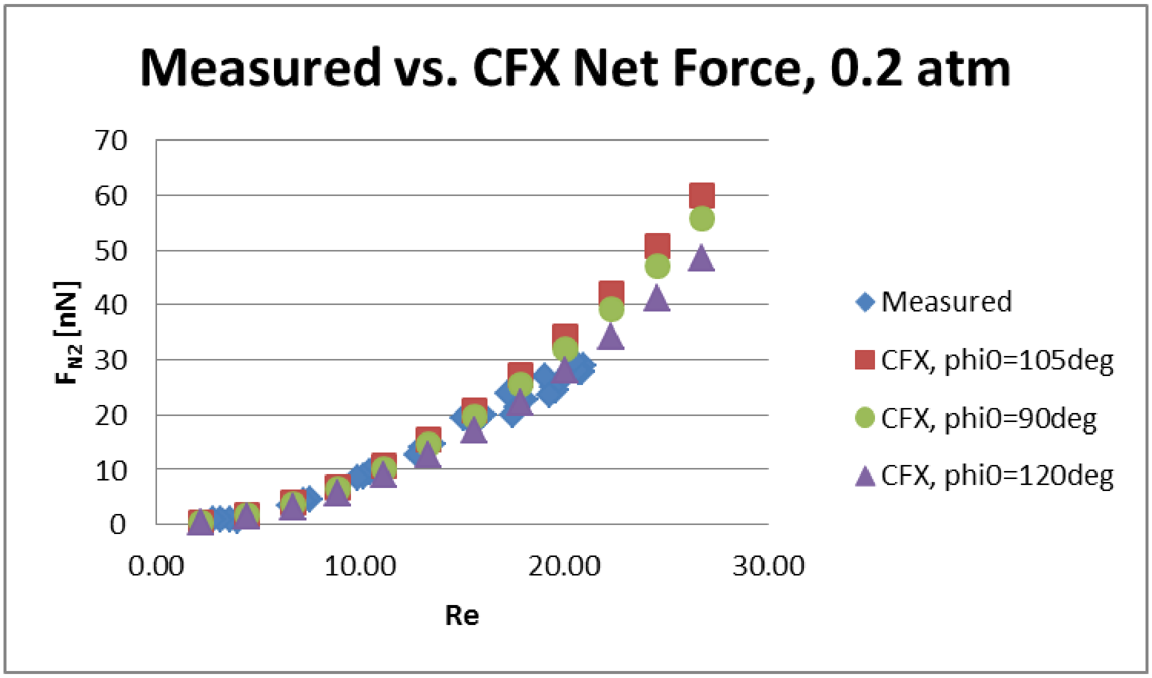


Figure 8-12: 0.2 atm N_2 measured vs. predicted force

The measured force from the torsion bar is consistent with the predictions from CFX. Several caveats are worth mentioning:

1. The wings of QSU8_17a were technically not pure cylindrical arcs.

As you can see in this picture, the tips of QSU8_17a are rotated out-of-plane a fair bit, likely due to mask misalignment during fabrication.

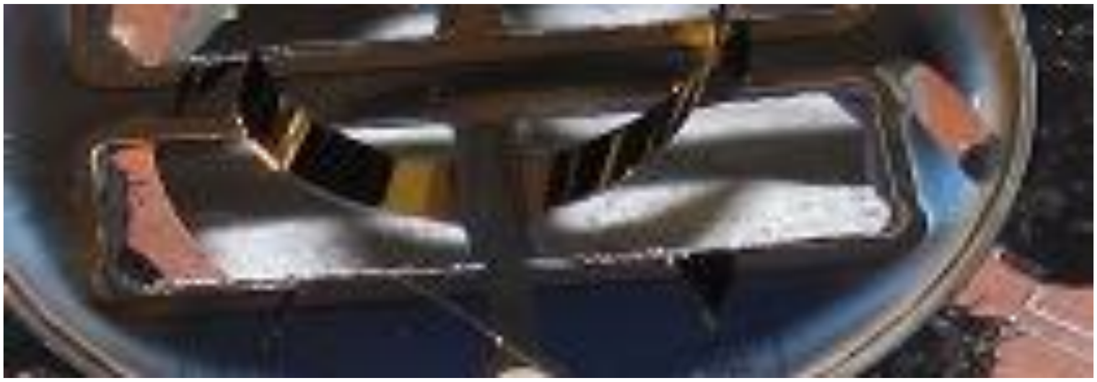


Figure 8-13: The tested wings twisted out of the plane of perfectly cylindrical arcs

The wings are not quite the same size as each other, nor do they have the same resonant frequency meaning that at the largest overall amplitude the wings are flapping out-of-phase with each other. Intuitively you might imagine that one wing flapping out of phase with the other would hinder the net force.

2. The force lost to the PCB is unclear.

Simulations predicted that interference from the PCB accounted for a 50-90% reduction in the measured force, but these simulation results were very sensitive to timestep and mesh density. The above plots assumed a 75% measured force loss for all measured values and this exaggerates the certainty in these measured values.

3. There is uncertainty in the wing amplitudes.

The wing amplitude in response to frequency changed somewhat over time, and furthermore measuring the amplitude entailed shining a moderate light on the wings which noticeably changed their curvature. This suggests that the actual wing amplitude and offset during the force measurements could be somewhat different from the ones measured in the amplitude tests under the same voltage and frequency conditions.

Section 8.5.2 The Air Force

These wings were also measured in air (rather than N_2), although not as extensively. In air at 200 V compared to N_2, the static bending angle ϕ_0 was lower ($86.7 \pm 1.9^\circ$ vs. $112 \pm 4^\circ$), as was the peak amplitude (0.15 vs. 0.17), and the resonant frequency (125 Hz vs. 130 Hz), as shown in Figure 8-14. These numbers, combined with the density and viscosity from Appendix L and summarized in Table 8-1, determine that the Reynolds number was lower (126 vs. 106) and the predicted net drag coefficient was slightly higher (0.05 vs. 0.045), for an overall lower predicted net force from two wings (0.156 μN vs. 0.185 μN).

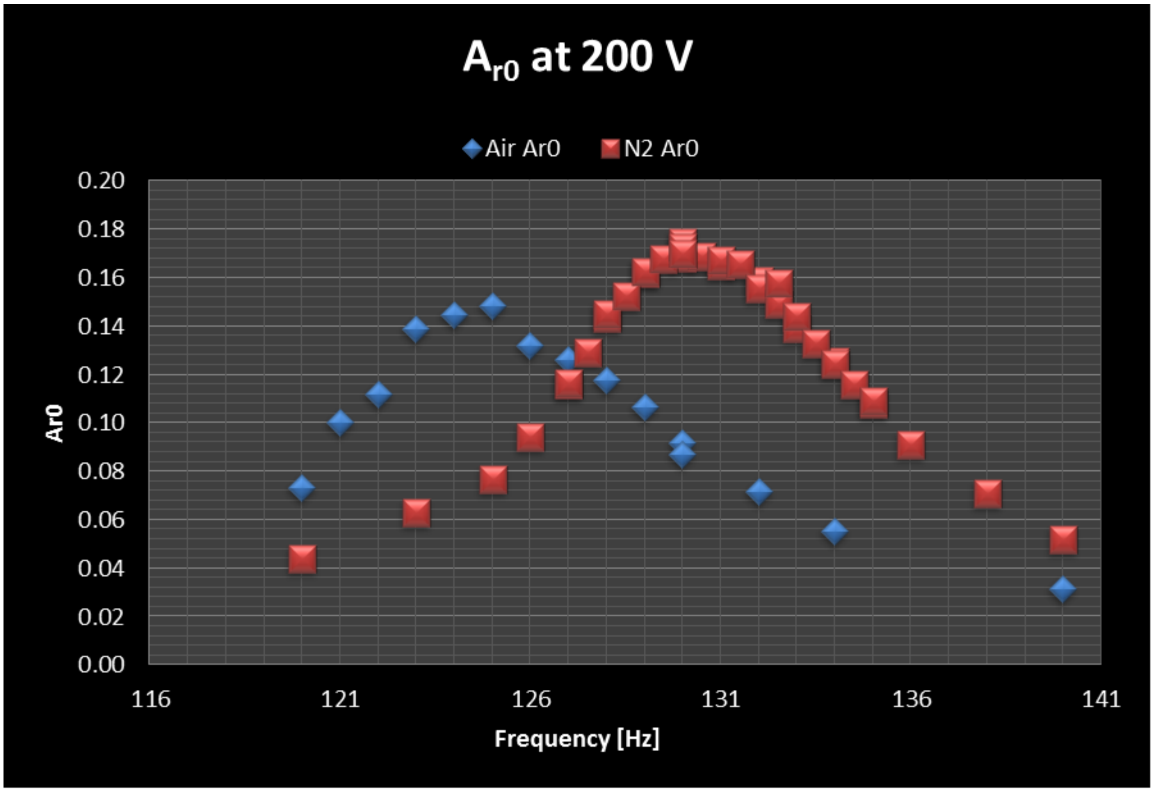


Figure 8-14: Measured flapping amplitudes in air and N_2

Table 8-1: Measures and predictions in air vs. N_2, using raw measured force

	ρ_{gas}	μ	A_{r0}	f	ϕ_0	f_{resCor}	Re	C_{DN}	F_{N2}	$F_{Measured}$
	[kg/m^3]	[μPa-s]		[Hz]	[^{o}]				[μN]	[μN]
N_2	1.165	17.4	0.17	130	112	1.08	126	0.045	0.185	0.063
air	1.205	18.3	0.15	125	105	1.07	106	0.05	0.156	0.12

However, the *measured* force in air was notably *higher* than in N_2, as shown in Figure 8-15, which shows the raw force measured by the torsion bar, without multiplying by 4 to account for the force lost to the PCB. This increased force in air relative to N_2 is surprising, and suggests a separate method of force loss besides the PCB. Recall from Figure 8-13 that these wings twisted at their tips when curved. The wings had a higher curvature and more out-of-plane twisting in N_2 than in air. Perhaps this twisting is responsible for reducing the peak force in N_2 so much lower than the peak force measured in air. If so, it illustrates the importance of symmetric fabrication to reduce wing twist, and gives another motive for favouring the wings with lower curvature.

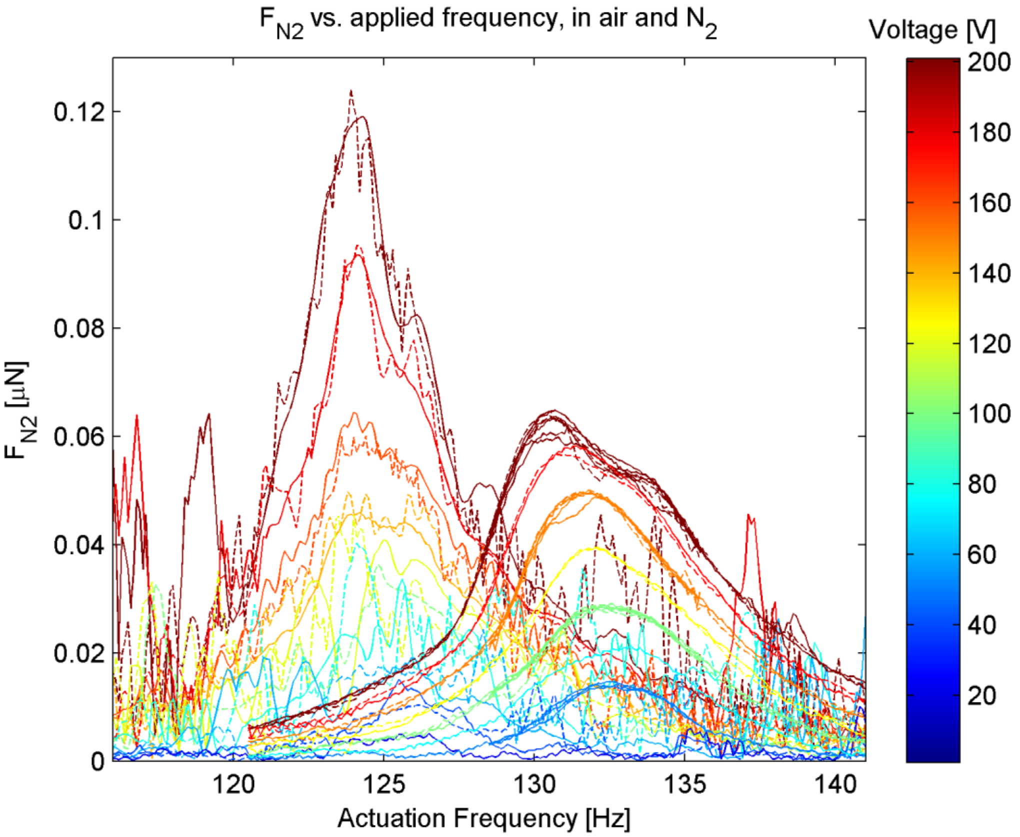


Figure 8-15: Raw measured force (i.e., not accounting for force lost to PCB) in air vs. in N_2

Section 8.6 Electrical parameters:

Do the BvD-predicted parameters match the measured ones given the electrical behaviour of the wings?   The easiest BvD-parameter to test is the package capacitance. Using equation (4.1) from Section 4.2.9.2, we find that each wing should have a package capacitance of 78 pF, and so the total test structure including bridge should have a capacitance of 183 pF. Measuring the test structure in air without the torsion wire attached gave a package capacitance of 180 pF, in good agreement with the prediction.

The torsion wire affects this capacitance measurement, and with the torsion wire attached and in vacuum, the total effective package capacitance was 351 pF.

The resonant branch parameters are more difficult to predict and test. As presented in Section 4.2.9.4 with equations (4.2) through (4.4), knowing the fluidic power dissipated and peak energy stored by the wings also determines what their electrical parameters should be in the BvD model:

$$R_1 = \frac{V^2}{2P_A},\ L_1 = \frac{2R_1^2}{V^2}U_{\max},\ C_1 = \frac{1}{\omega_1^2 L_1}$$

Chapter 2 told us the peak mechanical energy stored in the wing is related to its flexural rigidity *EI*, its length *L*, and its RS mode amplitude A_{r0} via equation (2.3),

$U_{\max} = \frac{1}{2}EI\left(\frac{\lambda^4}{4}\frac{A_{r0}^2}{L}\right) = \tfrac{1}{2}EI\frac{A_{r0}^2}{L}3.09059$, where $\lambda \approx 1.8751$ is the first root of $\cos\lambda\cosh\lambda + 1 = 0$. The RS mode amplitude at resonance is related to the applied voltage and quality factor through the amplification factor, $A_{r0} = QA_{r0DC} = Q\cdot\left(A_{r0DCperV}V\right)$, and the quality factor and peak mechanical energy stored determine rate of energy dissipation given the resonant frequency:

$$Q = \frac{2\pi U_{\max}}{TP_A} = \omega_1\frac{U_{\max}}{P_A} \to \frac{1}{P_A} = \frac{Q}{\omega_1 U_{\max}}$$

To test the electrical model we can use measured values from other sections to calculate what the BvD parameters should be, and then plot the current they would give against the actual measured current. We have to be careful which values to use though, because due to time constraints, detailed electrical measures were only taken in vacuum where the wing had a different resonant frequency and bending angle, indicating some wing properties had changed. In Section 8.1 we found A_{r0PerV} = 4.8e-5/V. This did not appear to be affected by the pressure at least down to 1e-4 atm, so although lower pressure increased the resonant frequency and static bending angle, the off-resonant transduction of voltage into bending stayed the same. This implies that vacuum may be decreasing the SU-8 density and increasing its built-in strain, but keeping its Young's modulus mostly the same.

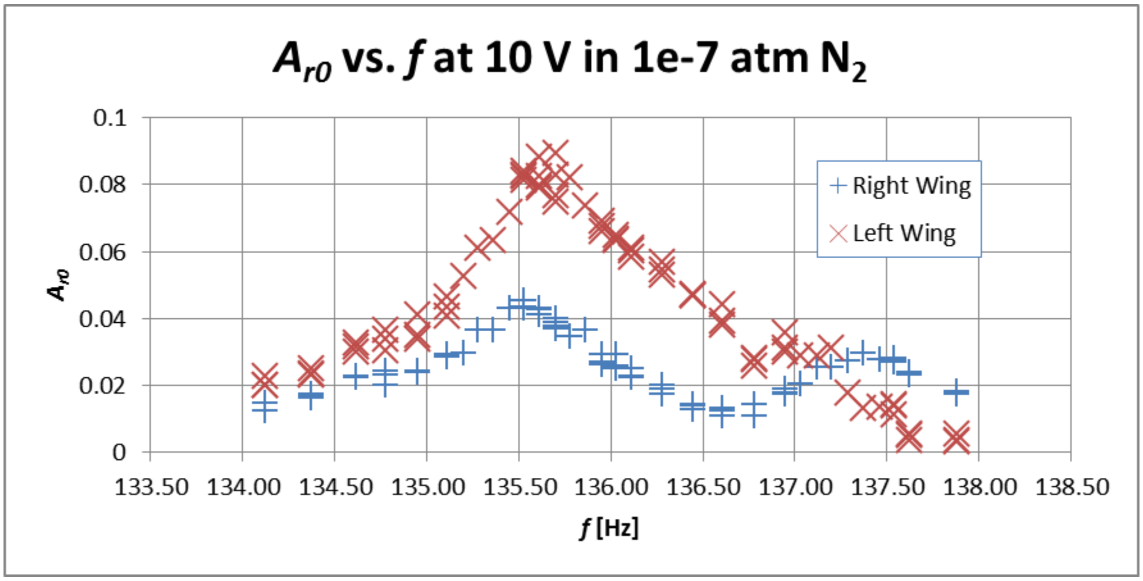


Figure 8-16: Measured amplitudes by-wing at 1e-7 atm during the 10 V sweep of electrical measurements

The camera amplitude measures in vacuum determined the resonant frequencies and resonant amplitudes shown in Figure 8-16. Off-resonance and at low frequency, a 10 V actuation signal produces an amplitude of 4.8e-4, so dividing the resonant amplitudes by 4.8e-4 determines the quality factor. Then, by using the known length (7.64 mm) and flexural rigidity (1.15e-8 Pa-m^4), we can calculate the BVD parameters in Table 8-2.

Table 8-2: Calculated BvD Parameters

	f_{res}	A_{r0}	$A_{r0DCPerV}$	Q	R	C
	[Hz]		[V^{-1}]		[MΩ]	[fF]
Right Wing	137.4	0.030	4.8e-5	62.5	1730	10.7
Left Wing	135.7	0.085	4.8e-5	177	617	10.7

Figure 8-17 plots the current magnitude and phase that were measured along with the ones predicted by these BvD model parameters for a 10 V amplitude signal in 10^{-7} atm N_2. The model does not completely agree with the measured current, but the general trend is in agreement.

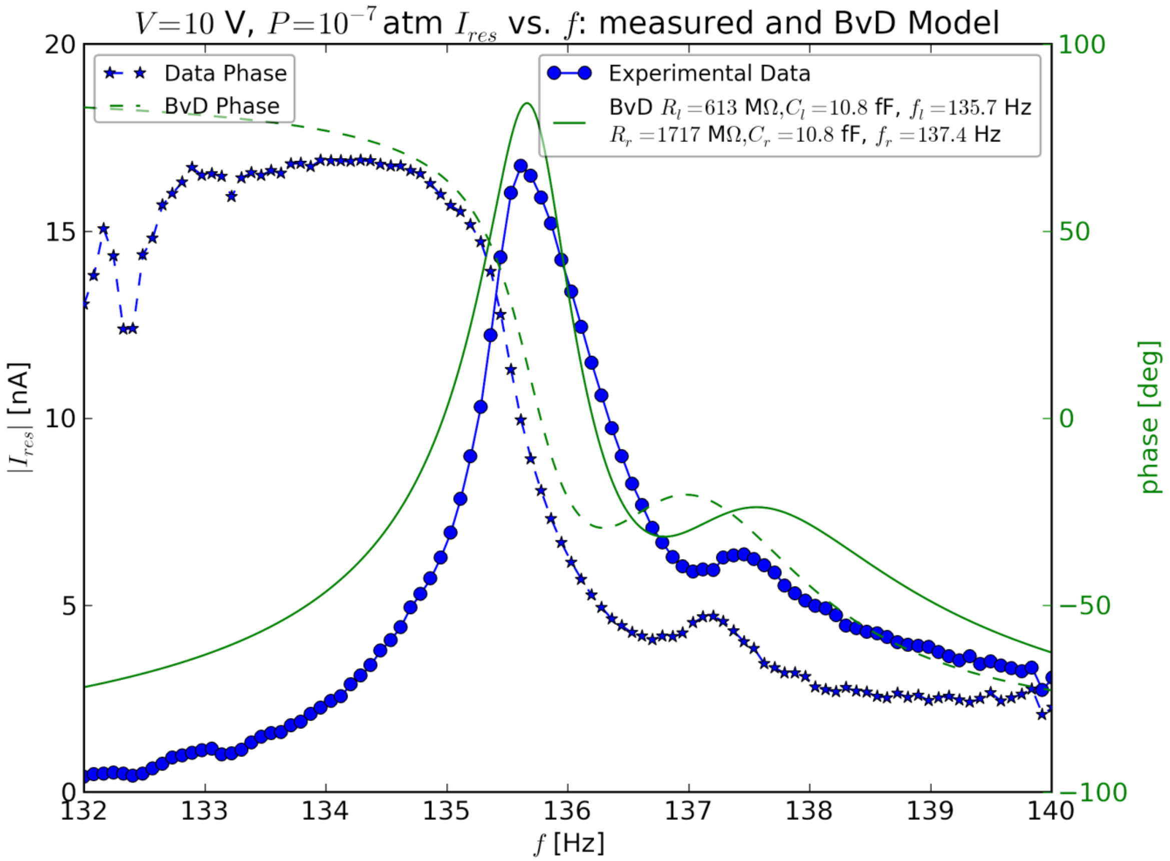


Figure 8-17: Experimental data vs. predicted BvD model at 10 V, using $A_{r0DCPerV}$ = 4.8e-5/V to determine quality factors

The BvD parameters plotted in Figure 8-17 hinged on the measurement of the $A_{r0DCPerV}$ from Section 8.1. This number has a significant amount of uncertainty associated with it, which is coupled to uncertainty in the SU-8 Young's modulus, and thus uncertainty in the wing's flexural rigidity, *EI*. Both the $A_{r0DCPerV}$ and *EI* have a strong effect on the BvD parameters, so determining the range of values from these will impact the output significantly. The high value for $A_{r0DCPerV}$ of 5.5e-5/V is associated with E_{Y_SU8} = 5.2 GPa and EI = 1.25e-8 Pa-m^4, while the low value of 4.1e-5/V is with E_{Y_SU8} = 3.3 GPa and EI = 1.07e-8 Pa-m^4. These lead to the BVD parameters in Table 8-3, and the plots in Figure 8-18 and Figure 8-19.

Table 8-3: Range of BvD parameters from range of input values

	f_{res}	A_{r0}	$A_{r0DCperV}$	EI	Q	R	C
	[Hz]		[V^{-1}]	[Pa-m^4]		[MΩ]	[fF]
Right Wing	137.4	0.03	4.8E-05	1.16E-08	62.5	1722	10.8
Left Wing	135.7	0.085	4.8E-05	1.16E-08	177.1	615	10.8
Right Wing	137.4	0.03	5.5E-05	1.25E-08	54.5	1388	15.3
Left Wing	135.7	0.085	5.5E-05	1.25E-08	154.5	496	15.3
Right Wing	137.4	0.03	4.1E-05	1.07E-08	73.2	2176	7.3
Left Wing	135.7	0.085	4.1E-05	1.07E-08	207.3	778	7.3

The range of BvD parameters in Table 8-3 is fairly large, but nevertheless insufficient to account for the plot data. Consider the best-guess BvD parameters plotted vs. some this electrical measure and another set of measures at the same voltage taken one week earlier, shown in Figure 8-20. At both the low and high-frequency end of the plot the measured current was lower than expected for the pure BvD model. This could reflect additional loss mechanisms off-resonance that were not accounted for using an RLC model.

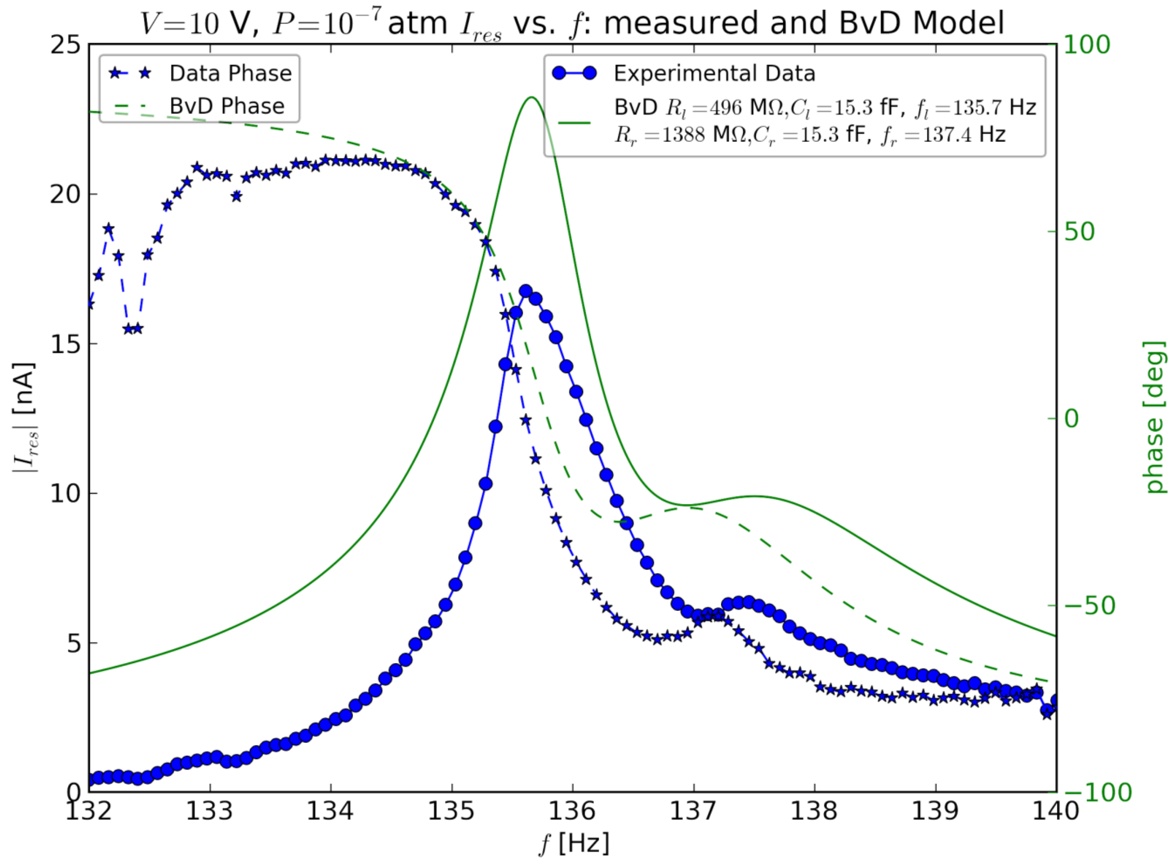


Figure 8-18: With both wings at $A_{r0DCPerV}$, EI = 5.5e-5, 1.25e-8

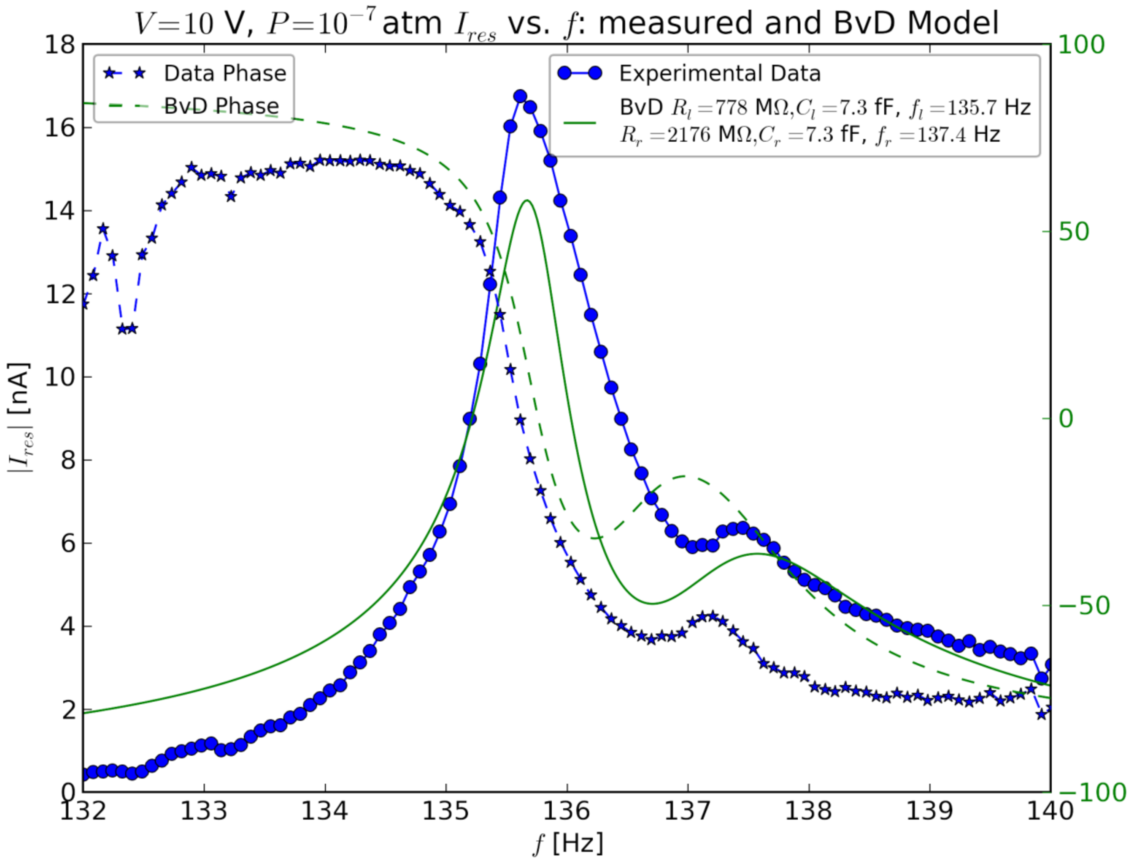


Figure 8-19: With both wings at $A_{r0DCPerV}$, EI = 4.1e-5, 1.07e-8

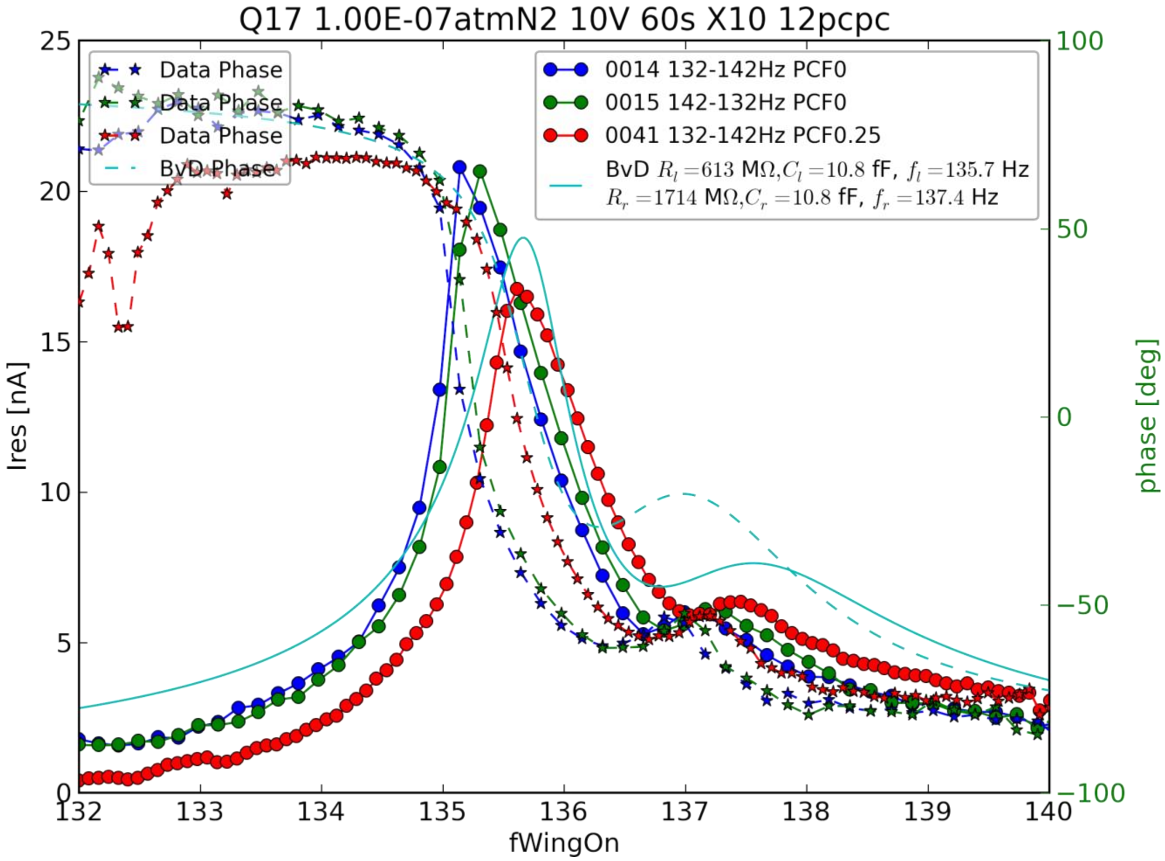


Figure 8-20: BvD model using best-guess parameters from the amplitude measures vs. three different electrical measures at 10V in vacuum, taken one week apart

At a more fundamental level than measurement uncertainty, the discrepancy may be related to the inability of the RLC resonator branch model to perfectly reflect what the flapping wing is doing. Referring to Figure 8-16, notice that when the left wing reaches its resonance at 135.7 Hz the right wing responds as well. This could be due to mechanical coupling between the two wings and is not covered by the BvD model: an RLC branch does not start storing more energy in its inductor and capacitor just because a branch in parallel enters resonance but the voltage applied across it stays the same.

Section 8.7   Summary

This section contained several related tests. If all of the analytical work is correct, then the *Q* measured by the BvD model and the *Q* measured by the amplitude should agree with each other. In Section 8.6 we found that this was generally the case on resonance, although the current off-resonance decayed faster than the BvD model predicted using linear damping. The magnitude of quality factor for the left wing was approximately 177, while that for the right wing was approximately 62. In contrast, the non-vacuum measurements from Section 8.4 found that for agreement with the CFX-predicted power-drag coefficients, the mechanical quality factor should be approximately 58. Looking more closely at Figure 8-6, it seems that the quality factor does increase with lower amplitude. The measures in vacuum may provide an answer: higher amplitude causes the wing resonances to overlap and interfere with each other, reducing the mechanical quality factor. If this is true, then predicted mechanical quality factor being closer to the right wing's quality factor makes sense.

The big picture of the measurements is that current measurement techniques have been limited by the inconsistent behaviour of the wings, and we can be fairly confident that this is due to inconsistent properties of SU-8. Nevertheless, all theoretical predictions for the wings were in agreement, and in agreement with each other, and the uncertainty on this only gives room for deviation within a factor of two. The electrical, visual, and force-based measures of the resonant frequency are in agreement, and the measured amplitude and force are consistent with both the net and power drag coefficients and the analytical framework, both in their general trends and in their absolute magnitude.

Chapter 9 Conclusion

Understanding that the flight mechanism is the least explored component for microrobotic flight, and assuming that MEMS parallel processing techniques are essential for mass-producing the future's microrobots, this work set out to explore a very simple wing design that permits easy MEMS fabrication: resonating curved cantilevers. I determined the large-amplitude resonant shape and frequency relations of these cantilevers, and presented equations to generate them. This shape agrees with experiments performed here, and its frequency relation agrees with simulations and published results. I performed thousands of CFD simulations to determine the forces and dissipated power from this resonant shape, and extended the drag coefficients of steady-state fluid mechanics to explain the average results over many cycles of motion. Confirming the scaling laws necessary to extend the fluid results using similarity of flow for equivalent Reynolds number situations, I developed an analytical framework to interpret the fluid mechanics results in the context of a complete microrobot. Using this analytical framework, I employed an optimization routine to determine that this flight method does indeed lead to flying robots, and how to optimize material choice and flapping dimensions for various applications. Next, I studied and optimized RIE thinning of x-cut quartz, and used this to fabricate 15 μm-thick Quartz:SU8 wings. To test the wings, I built a torsion bar, and combined it with a camera and lock-in amplifier to measure the wings' flapping amplitude, force generation, and electrical parameters in tandem, all in various ambient gas pressure conditions. Finally, I analyzed the experimental results, and demonstrated that they are consistent with the theory for the resonant shape, fluid mechanics results, and the analytical framework.

To the best of my knowledge, the smallest flight mechanism for flying robots that has been published with a net force is the Harvard RoboBee whose flight mechanism (wings + actuators + mechanical gain components) has a mass of 60 mg [64]. In contrast, the fabricated flight mechanism in this work has a mass of only 1 mg, and the analytical framework determined the performance of even smaller robots. These Quartz:SU8 wings generated a force of 200 nN when flapping in 1 atm N_2 at their 130 Hz resonance and 200 V-induced amplitude of $A_{r0} = 0.165$. Although this force was only 2% of their weight, it agreed with the fluid mechanics predictions for this amplitude and Reynolds number. Furthermore, the default curvature, bending amplitude, and resonant frequency agreed with the predictions from the analytical framework and fluid mechanics simulations. That same framework suggests that building an identical wing using silicon rather than SU-8 should significantly improve the design. Using Si instead of SU-8 doubles the resonant frequency and lowers the mechanical damping, which quadruples the amplitude at a lower input voltage, and increases the Reynolds number by a factor of 8 (for the same wing dimensions). Furthermore, the fluid mechanics simulations indicate this increase in Reynolds number and amplitude would push the wings into a much more favourable flight regime, increasing their net drag coefficient by a factor of 5. Together, this change in Reynolds number and drag coefficient increase the net force output by a factor of 320, enough for the wings to lift 6 times their own mass.

This is indeed a promising picture for this form of MEMS-fabricated microrobotic flight, but it certainly is not the end of the story. Additional work needs to be done even in testing the rectangle wings simulated in this thesis. Fabricating wings with tighter tolerances to create shapes closer to pure cylinders will allow better testing of the fluid force predictions, and less interference from close-but-unequal resonances. Making wings out of materials which are more resistant to pressure changes (i.e., Si rather than SU-8) will enable extending vacuum measures of mechanical damping to higher pressures, and thus will allow us to more carefully test the power drag coefficient. Making the wings out of materials with lower intrinsic mechanical damping should test whether material choice can reduce the wings' mechanical damping and allow larger amplitudes at lower voltages. Constructing test structures with smaller PCBs that have more clearance for the flapping wings and less drag force themselves will reduce their interference with the wing force measurement, greatly increasing its accuracy.

Rectangles with a length-to-width ratio of 4 are not the only wings permitted by MEMS fabrication. Future work on the flight mechanism should explore other potential fluid diode wings, including different rectangles (Length-to-width = 5?), streamlined shapes more similar to insect wings, grids of flaps, textured surfaces, and wings with a lip around their edges in the thickness direction, to name a few. Much additional work is also possible in material selection. The Quartz:Si, Quartz:SU-8, and InGaP wings analyzed here are barely a scratch on the surface of material possibility. The optimization routine employing the analytical framework, especially if enhanced with specific application requirements, can do much to enable material optimization.

So what is the outlook for the future? Is it possible to create autonomous robots on the scale of a fruit fly fabricated in parallel by MEMS technology? Much work certainly remains, but we are on our way to answering that question, and, at least so far, the answer is yes.

Appendix A Friction and Form Drag in Thrust and Lift Flight: A MEMS Scaling Exercise

One way to model the drag force is by separating out the contribution due to pressure ("form drag", dominates on large scales) and the contribution due to viscosity ("[skin] friction drag", dominates on small scales). To do this and still use drag coefficients in the usual formula for drag force, $F_D = \frac{1}{2}\rho C_D A v^2$, we need to break the drag coefficient into separate parts due to friction and form drag: $C_D = C_{Dfric} + C_{Dform}$

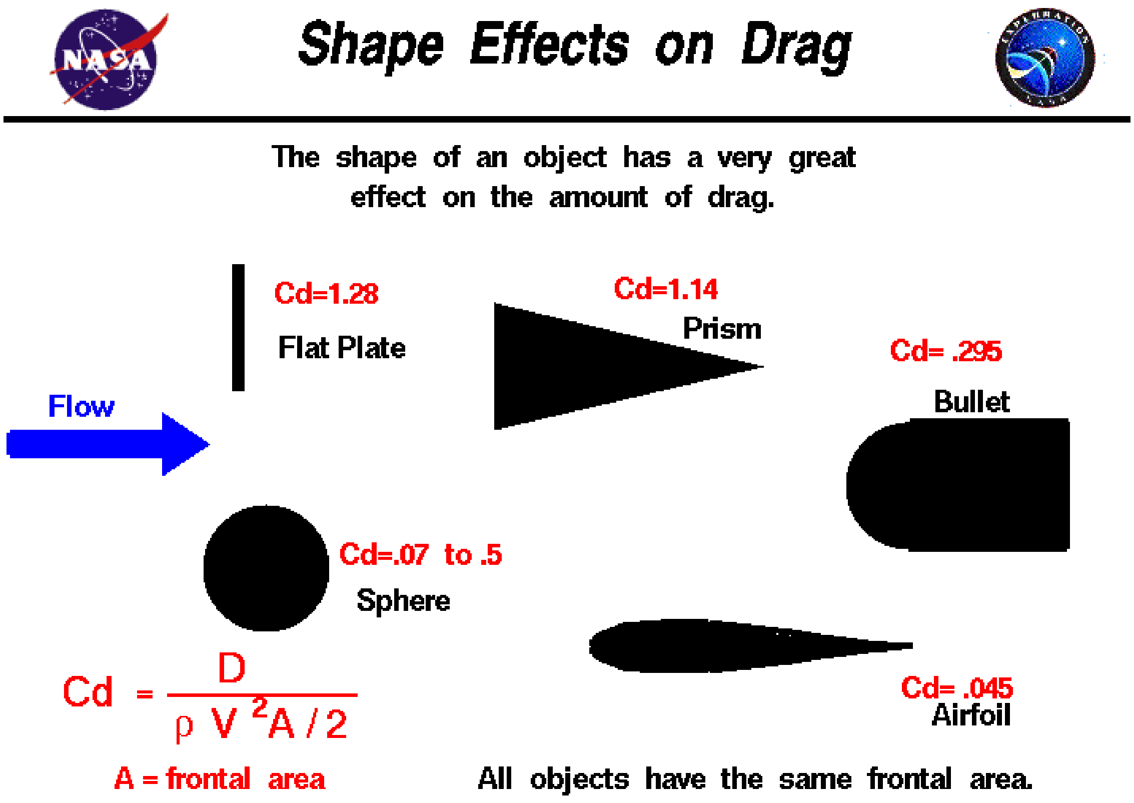
In an ideal case, the friction drag coefficient is largely independent of shape and inversely proportional to the Reynolds number, as in the Stokes drag formula for the drag coefficient of a sphere:

$$C_{Dfric} = \frac{24}{\mathrm{Re}} \quad \text{(A.1)}$$

where $\mathrm{Re} = \rho \frac{vL}{\mu}$ taking v as the free-stream velocity and L as the sphere's diameter. In contrast, form drag is independent of Reynolds number and highly shape-dependent:

$$C_{Dform} = C_{shape}$$

Figure A-1 shows form drag coefficient of common shapes, which ranges from about 0.05 to 1.3. For a flapping wing, the skin friction drag force (on average) cancels out, so it contributes only to the power dissipation (and not the net force), and all of the useful force comes from the form drag.



Shape Effects on Drag

The shape of an object has a very great effect on the amount of drag.

$$Cd = \frac{D}{\rho\ V^2 A/2}$$

A = frontal area

All objects have the same frontal area.

Figure A-1: Form drag coefficients are greatly affected by shape, from [65]

Let L be the length and ω be the flapping frequency. Define the net drag coefficient C_{DN} as the effective drag coefficient difference on the up and downstroke so that the thrust force scales like:

$$F_T = \tfrac{1}{2}\rho_{air} C_{DN} A v^2$$
$$F_T \propto C_{DN} L^4 \omega^2$$

To a first approximation, since it's due to form drag, the net drag coefficient should be largely independent of Reynolds number.

Next, define the power drag coefficient C_{DP} so that the effective power used to hover scales like:

$$P_A = \text{avg}\left(\mathbf{F}\cdot\mathbf{v}\right) \equiv \tfrac{1}{2}\rho_{air} C_{DP} A v^3$$
$$P_A \propto C_{DP} L^5 \omega^3$$

With this definition, the power drag coefficient should be from a combination of form drag and friction drag. Defining the Reynolds number as $\mathrm{Re} \equiv \frac{Lv}{\rho} \propto L^2\omega$, the power-drag coefficient is

$$C_{DP} = C_{DPfriction} + C_{DPform}$$
$$= \frac{C_1}{L^2\omega} + C_2$$

meaning that the average power used is highly size-dependent:

$$P_A \propto \left(\frac{C_1}{L^2\omega} + C_2\right)L^5\omega^3 = C_1L^3\omega^2 + C_2L^5\omega^3$$

Section A.1 Thrust flight:

For maintaining altitude using thrust flight, set: thrust = weight

Since weight scales with L^3, setting the thrust force to directly oppose gravity relates the frequency to the wing length:

$$F_T = W$$
$$L^4\omega^2 \propto L^3$$
$$\therefore \omega \propto L^{-0.5}$$

Substitute the frequency into the power used equation to find:

$$P_A = C_1'L^2 + C_2L^{3.5}$$

Section A.2 Lift Flight:

In lift flight, there are two force balance equations:

1) Lift-Weight balance: $F_{Lift} = F_{Weight}$

2) Thrust-Drag balance: $F_T = F_{BodyDrag}$

Section A.2.1 Lift-Weight Balance:

The lift force is given by $F_{Lift} = \rho_{air} v_{fwd}^2 \pi LB\alpha$, where α is the angle of attack and B is the chord length. i.e., $F_{Lift} \propto L^2 v_{fwd}^2$.

Setting the lift force equal to the weight gives a scaling on the required forward velocity for lift-flight:

$$F_{Lift} = F_{Weight}$$
$$\therefore L^2 v_{fwd}^2 \propto L^3$$
$$\rightarrow v_{fwd} \propto \sqrt{L}$$

Section A.2.2 Thrust-Drag Balance:

The thrust is the same as above:

$$F_T = \tfrac{1}{2} \rho_{air} C_{DN} A v^2$$
$$F_T \propto C_{DN} L^4 \omega^2$$

The body drag is given by the drag equation from the earlier power calculation:

$F_D = \tfrac{1}{2} C_{DBody} A v_{fwd}^2 \propto C_{DBody} L^2 v_{fwd}^2$, where the body drag coefficient is

$$C_{DBody} = C_{DBfriction} + C_{DBform}$$
$$= \frac{C_{B1}}{L v_{fwd}} + C_{B2}$$

giving $F_{DBody} \propto C_{B1} L v_{fwd} + C_{B2} L^2 v_{fwd}^2$.

Substituting $v_{fwd} \propto \sqrt{L}$ gives that the drag force is

$$F_{DBody} = C_{B1}' L^{1.5} + C_{B2}' L^3$$

Equate this to the thrust force to find the new frequency-length relation for lift flight:

$$C_{DN}L^4\omega^2 \propto C_{B1}'L^{1.5} + C_{B2}'L^3$$
$$\therefore \omega \propto \sqrt{C_{B1}'L^{-2.5} + C_{B2}'L^{-1}}$$

And finally the power used to do it:

$$P_A \propto \left(\frac{C_1}{L^2\omega} + C_2\right)L^5\omega^3 = C_1L^3\omega^2 + C_2L^5\omega^3$$
$$\therefore P_{ALiftFlight} = C_{1P}'L^3\left(C_{B1}'L^{-2.5} + C_{B2}'L^{-1}\right) + C_{2P}'\left(C_{B1}'L^{-2.5} + C_{B2}'L^{-1}\right)^{1.5}$$

Scaling domains:

1) L is very large; form drag dominates forward motion & thrust mechanism:

   $\omega \propto \sqrt{0 + C_{B2}'L^{-1}} \propto L^{-0.5}$ and $P_{ALiftFlight} \propto 0 + C_2L^5\omega^3 \propto L^{3.5}$

2) L is moderately large; friction drag dominates forward motion, but form drag dominates thrust mechanism $\omega \propto \sqrt{C_{B1}'L^{-2.5} + 0} \propto L^{-1.25}$ and

   $P_{ALiftFlight} \propto 0 + C_2L^5\omega^3 \propto L^{1.25}$

3) L is very small; friction drag dominates forward motion & thrust mechanism:

   $\omega \propto \sqrt{C_{B1}'L^{-2.5}} \propto L^{-1.25}$ and $P_{ALiftFlight} \propto C_1L^3\omega^2 + 0 \propto C_1L^{0.5}$

(Note that the opposite middle region, friction drag for thrust mechanism & form drag for body, is nonphysical, as it requires the wing velocity to be slower than the bird velocity.)

Section A.3 Summary and Comparison

Lift flight power required:

~L$^{3.5}$ for large objects

~L$^{1.25}$ for mid-sized objects

~L$^{0.5}$ for small objects

Thrust flight power required:

~L$^{3.5}$ for large objects

~L^2 for small objects

The available power for biological systems and fossil-fuel aircraft is volume based: $P_{Avail} \propto L^3$. Defining "Suitability" as the available power divided by the required power allows comparing which size scales we expect each flight design to dominate on.

Using reasonable (but highly variable!) numbers for parameters, we can get some idea for where this transition area should be. The Maple program in section Section A.4 calculates the suitability vs. length scale for volume-based power, area-based power, and a combination of both vs. length scales, generating Figure A-2 through Figure A-4.
As expected, volume-based power supplies favour larger designs, area-based ones favour smaller ones, and a hybrid power supply works over a broader range. Furthermore, thrust flight dominates lift flight on smaller size scales but for larger objects lift flight wins out.

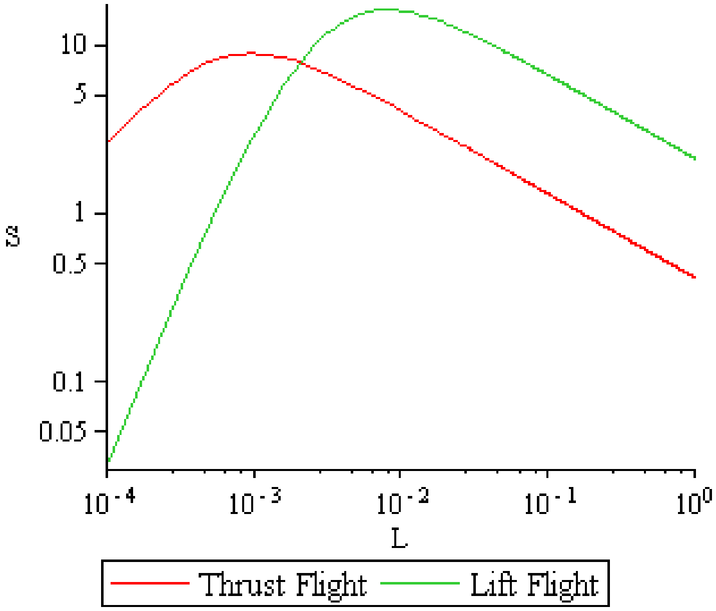


Figure A-2: Suitability vs. Length [m] for a volume-based power supply

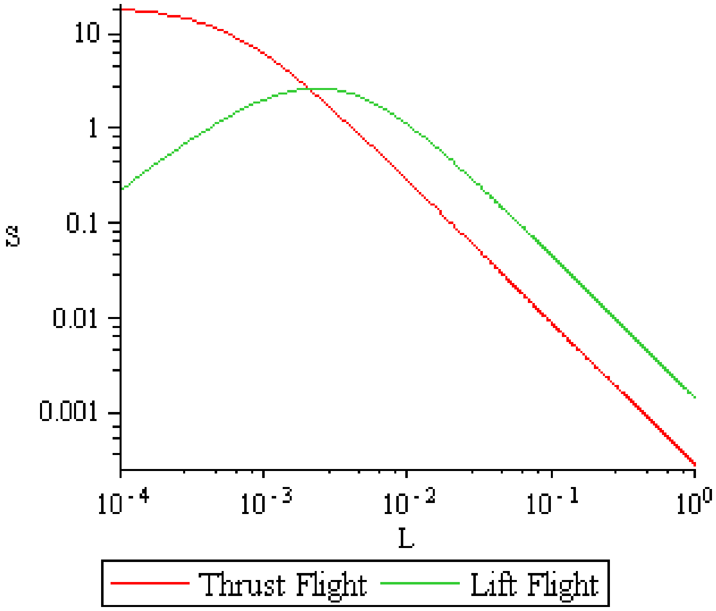


Figure A-3: Suitability vs. Length [m] for an area-based power supply

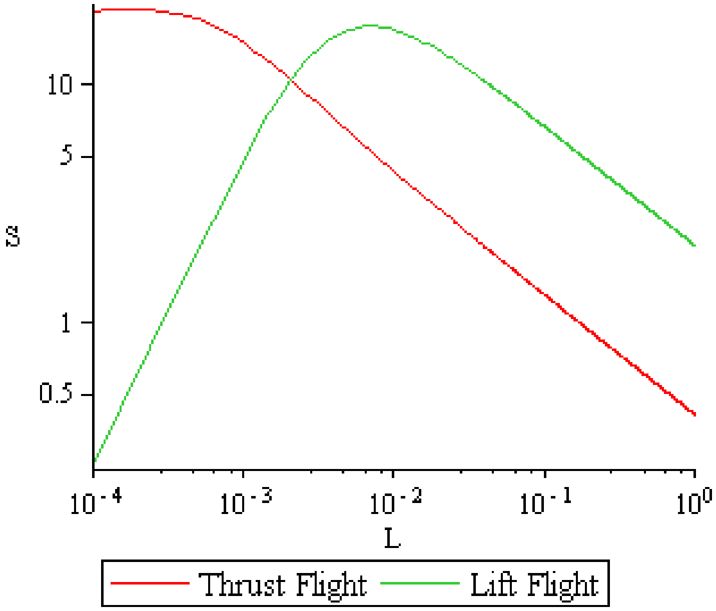


Figure A-4: Suitability vs. Length [m] for volume- and area-based power supplies

Section A.4 Maple Program to Derive Thrust & Lift Flight Plots

```
> restart;
> rho_air:=1.22: nu:=1.46e-5:
> alpha:=evalf(15*Pi/180);
```

$$\alpha := 0.2617993878$$

```
> c:=L/4;
```

$$c := \frac{L}{4}$$

```
> FL:=evalf(2*rho_air*vfwd^2*L*c*Pi*alpha);
```

$$FL := 0.5017048906\ vfwd^2 L^2$$

```
> Cdfric:=R->24/R;
```

$$Cdfric := R \rightarrow \frac{24}{R}$$

```
> with(plots):
> loglogplot(Cdfric(R)+0.2, R=0.1..1e6):
> theta:=A*sin(2*Pi*f*t);
```

$$\theta := A \sin(2\,\pi\, f\, t)$$

```
> omega:=diff(theta, t);
```

$$\omega := 2\, A \cos(2\,\pi\, f\, t)\,\pi\, f$$

```
> A:=0.9*Pi/2;
```

$$A := 0.4500000000\ \pi$$

```
> RMSomega:=simplify(sqrt(f*int(omega^2, t=0..1/f)),
assume='positive');
```

$$RMSomega := 6.280977780\ f$$

```
> CshapeWing:=1;
```

$$CshapeWing := 1$$

```
> Ft:=1/2*2*int(rho_air*CshapeWing*c*(RMSomega*r)^2/2,
r=0..L); int(r^2, r=0..L);
```

$$Ft := 2.005409662\ L^4 f^2$$

$$\frac{L^3}{3}$$

```
>
P:=2*(CshapeWing+2*Cdfric(sqrt(L*c)*(2/3*L*RMSomega)/nu))*In
t(rho_air*c*(RMSomega*r)^3/2, r=0..L);
```

$$P := 2\left(1 + \frac{0.0003347249542}{\sqrt{L^2}\ L\, f}\right)\int_0^L 37.78780057\ L\, f^3\, r^3\, dr$$

```
> P:=simplify(value(P), assume='positive');
```

$$P := 0.3778780056\ 10^{-11}\,(0.5000000000\ 10^{13}\, L^2\, f + 0.1673624771\ 10^{10})\, L^3\, f^2$$

```
> Cg:=9.81*5e-7/(1.7e-3)^3;
```

$$Cg := 998.3716670$$

```
> Fw:=L->Cg*L^3;
```

$$Fw := L \to Cg\, L^3$$

Lift Flight

```
> FDWings:=2*(.1)*rho_air*c*L*(vfwd)^2/2 +
2*Cdfric(vfwd*sqrt(c*L)/nu) * rho_air*c*L*(vfwd)^2/2;
```

$$FDWings := 0.03050000000\ vfwd^2\, L^2 + \frac{0.0002137440000\ vfwd\, L^2}{\sqrt{L^2}}$$

```
> FDBody:=(.1)*rho_air*Pi*c^2/4*(vfwd)^2/2 +
Cdfric(vfwd*sqrt(Pi*c*L)/nu) * rho_air*Pi*c*L*(vfwd)^2/2;
```

$$FDBody := 0.0009531250000\ \pi\, L^2\, vfwd^2 + \frac{0.0001068720000\ vfwd\, \pi\, L^2}{\sqrt{\pi\, L^2}}$$

```
> EqnThrustDrag:=simplify(0.25*Ft=FDWings+FDBody,
assume='positive');
```

$$EqnThrustDrag :=$$
$$0.5013524155\ L^4\, f^2 = 0.03349433050\ vfwd^2\, L^2 + 0.0004031696880\ vfwd\, L$$

```
> EqnLiftWeight:=0.5*FL=Fw(L);
```

$$EqnLiftWeight := 0.2508524453\ vfwd^2\, L^2 = 998.3716670\ L^3$$

```
> solve(EqnLiftWeight, vfwd): LFvfwd:=simplify(%[1],
assume='positive'); vFlap:=2/3*L*RMSomega;
```

$$LFvfwd := 63.08657564\ \sqrt{L}$$
$$vFlap := 4.187318520\ L\, f$$

```
> subs(vfwd=LFvfwd, EqnThrustDrag);
```

$$0.5013524155\ L^4\, f^2 = 133.3046227\ L^3 + 0.02543459502\ L^{(3/2)}$$

```
> solve(subs(vfwd=LFvfwd, EqnThrustDrag), f):
LFf:=simplify(%[1], assume='positive');
vFlapLF:=simplify(subs(f=LFf, vFlap), assume='positive');
#semilogplot(abs(LFf), L=1e-5..1);
evalf((1.8e10/8.5e13)^(2/3));
loglogplot([LFvfwd, vFlapLF], L=1e-5..100, numpoints=1000):
```

$$LFf := \frac{0.00001410398673\ \sqrt{0.1336651892\ 10^{13}\, L^{(3/2)} + 0.255033913\ 10^9}}{L^{(5/4)}}$$

$$vFlapLF := \frac{0.00005905788484\ \sqrt{0.1336651892\ 10^{13}\, L^{(3/2)} + 0.255033913\ 10^9}}{L^{(1/4)}}$$

```
> PLF:=0.5*simplify(subs(f=LFf, P), assume='positive');
```

$$PLF := 0.5449710125\ 10^{-21}\, ($$
$$0.48634437\ 10^8\, L^{(3/4)}\, \sqrt{0.1336651892\ 10^{13}\, L^{(3/2)} + 0.255033913\ 10^9}$$
$$+ 0.1154223980\ 10^{10})\, \sqrt{L}\ (0.1336651892\ 10^{13}\, L^{(3/2)} + 0.255033913\ 10^9)$$

Thrust Flight

```
> EqnThrustWeight:=Fw(L)=Ft;
```

$$EqnThrustWeight := 998.3716670\ L^3 = 2.005409662\ L^4 f^2$$

```
> solve(EqnThrustWeight, f); TFf:=%[1];
```

$$\frac{22.31231190}{\sqrt{L}}, -\frac{22.31231190}{\sqrt{L}}$$

$$TFf := \frac{22.31231190}{\sqrt{L}}$$

```
> PTF:=simplify(subs(f=TFf, P), assume='positive');
```

$$PTF := 0.1881225075\ 10^{-8} (0.1115615595\ 10^{15} L^{(3/2)} + 0.1673624771\ 10^{10}) L^2$$

```
> loglogplot([TFf, LFf], L=1e-5..100, legend=["Thrust Flight
", "Lift Flight "]):
> PaBatt := 72.2*L^3*1200:
PaSolar := 120*2*c*L:

> Pa := PaBatt+PaSolar:
loglogplot([Pa/PTF, Pa/PLF], L=1e-4..1, legend=["Thrust
Flight", "Lift Flight"], labels=["L","S"], numpoints=1000):

> Pa := PaBatt:
loglogplot([Pa/PTF, Pa/PLF], L=1e-4..1, legend=["Thrust
Flight", "Lift Flight"], labels=["L","S"], numpoints=1000):

> Pl:=2*(CshapeWing*int(rho_air*c*(sqrt(RMSomega^2)*r)^3/2,
r=0..L) +
2*Cdfric(sqrt(L*c)*(2/3*L*sqrt(RMSomega^2+vfwd^2))/nu)*int(r
ho_air*c*(sqrt(RMSomega^2+vfwd^2)*r)^3/2, r=0..L));
```

$$Pl := 18.89390028\ L^5 (f^2)^{(3/2)} + \frac{0.0001603080000\ L^4 (39.45068187\ f^2 + vfwd^2)}{\sqrt{L^2}}$$

```
> PLF2:=simplify(subs([f=LFf, vfwd=LFvfwd], Pl),
assume='positive'):
```

$$\begin{aligned} PLF2 := {} & 0.2000000000\ 10^{-13} \sqrt{L}\ (\\ & 0.3542710078\ 10^{13} L^{(9/4)} \sqrt{0.1336651892\ 10^{13} L^{(3/2)} + 0.255033913\ 10^9} \\ & + 0.675951023\ 10^9 L^{(3/4)} \sqrt{0.1336651892\ 10^{13} L^{(3/2)} + 0.255033913\ 10^9} \\ & + 0.8407789170\ 10^{14} L^{(3/2)} + 0.1604210778\ 10^{11} + 0.3190061892\ 10^{14} L^{(7/2)}) \end{aligned}$$

```
> loglogplot([Pa/PTF, Pa/PLF,Pa/PLF2], L=1e-5..10,
legend=["Thrust Flight Suitability", "Lift Flight
Suitability", "Total v Lift Flight Suitability"],
labels=["L","S"]):
```

Appendix B MATLAB Program to explore Multimorph Frequency

```
t=[10.1e-6, 7e-6];  %Layer thickness; layer 1, layer 2
E=[86.7e9, 4.4e9];  %Layer young's modulus
rho=[2600,1190]; %Layer density
rhom=sum(rho.*t)/sum(t); %weighted average beam density
B=2e-3; %Beam width
A=t*B;  %Layer Cross-sectional area
L=8e-3; %Beam length

%%Formulation 2 (multimorph)
y=0;
for(i=1:length(t))
    y(i)=sum(t(1:(i-1))) + t(i)/2; %yi is the coordinate of the centroid of the ith layer
end
Ybar=sum(A.*E.*y)/sum(A.*E);
I0=B*t.^3/12;
EI=sum(E.*I0 + A.*E.*(y-Ybar).^2)
fres=1.875^2/L^2/(2*pi)*sqrt(EI/B/sum(t)/rhom)

%%Formulation 1 (only works for 2 layers); extension of Timoshenko
m=t(1)/t(2);
n=E(1)/E(2);
Ey=m*E(1)/((m+1)^3*(m*n+1))*(4+6*m+4*m^2+n*m^3+1/(n*m));
I=B*(sum(t))^3/12;
EI1=Ey*I;
fres1=1.875^2/L^2/(2*pi)*sqrt(EI1/B/sum(t)/rhom)
```

Appendix C MATLAB program to explore Multimorph Curvature

(This is MATLAB code)

```
clear all;
close all;

t=[10.1e-6, 7e-6];  %Layer thickness; layer 1, layer 2
E=[86.7e9, 4.4e9];  %Layer young's modulus
rho=[2600,1190]; %Layer density
rhom=sum(rho.*t)/sum(t); %weighted average beam density
B=2e-3; %Beam width
A=t*B;  %Layer Cross-sectional area
L=8e-3; %Beam length
epfree=[-0.0016446, -0.00495-0.00715];
nL=length(t); %number of layers

%%Formulation 2 for res frequency(works for multimorphs)

y=0;
for(i=1:length(t))
    y(i)=sum(t(1:(i-1))) + t(i)/2; %yi is the coordinate of the centroid of the ith layer
end
Ybar=sum(A.*E.*y)/sum(A.*E);
I0=B*t.^3/12;
EI=sum(E.*I0 + A.*E.*(y-Ybar).^2);
fres=1.875^2/L^2/(2*pi)*sqrt(EI/B/sum(t)/rhom);
%DeVoe formulation; from the DeVoe Paper
Ddv=1/sum(E.*I0)*y
Adv=zeros(nL);
for(i=1:nL-1)
    %Generate the 2 nonzero entries in Row i:
    Adv(i,i) = 1/(A(i)*E(i));
    Adv(i,i+1) = -1/(A(i+1)*E(i+1));
end
for(i=1:nL)
    %Generate the row of 1s in the last row of A:
    Adv(nL,i)=1;
end
Bdv=[t(1:nL-1)+t(2:nL),0]';
Cdv=[epfree(2:nL)-epfree(1:nL-1), 0]';
cdv=Ddv*Adv^-1*Cdv/(1-0.5*Ddv*Adv^-1*Bdv)
phiDegdv=cdv*L*180/pi
```

```
%%Formulation 1 for res frequency (only works for 2 layers)
m=t(1)/t(2);
n=E(1)/E(2);
Ey=m*E(1)/((m+1)^3*(m*n+1))*(4+6*m+4*m^2+n*m^3+1/(n*m));
I=B*(sum(t))^3/12;
EI1=Ey*I;
fres1=1.875^2/L^2/(2*pi)*sqrt(EI1/B/sum(t)/rhom);
%%Timoshenko Formulation for curvature:
%c = (strain difference)/h * curvrel, and:

DelEp=(epfree(2)-epfree(1));
h=sum(t);
cTimoshenko=6*DelEp*(1+m)^2/((m^2+1/n/m)*(1+n*m)+3*(1+m)^2)/h
phiDegTimoshenko=cTimoshenko*L*180/pi
```

Appendix D Resonant derivation of a non-curved cantilever: Euler-Bernoulli Beam Theory

Following is a summary of using Euler-Bernoulli beam theory to determine the dynamic equation of a straight beam; as presented in [25].

Flexural rigidity = stiffness = EI

$EI\frac{d^4w}{dx^4}=q(x)$, where $w(x)$ is the transverse displacement of the beam at point x from the no force condition due to the distributed load q (force per unit length).

Derivatives of w have important physical meanings:

$\frac{dw}{dx}$ is the slope of the beam.

$M=-EI\frac{d^2w}{dx^2}$ is the bending moment of the beam.

$Q=-\frac{d}{dx}\left(EI\frac{d^2w}{dx^2}\right)$ is the shear force in the beam.

Section D.1 Deriving the bending moment equation

In actuality, $M=-EI\frac{1}{\rho}$, where ρ is the radius of curvature of the beam. When the bending is small, then the curvature is $\frac{1}{\rho}\approx\frac{d^2w}{dx^2}$.

--> Restoring moment of the beam is linear in displacement from neutral curvature!

$\boxed{M=-EI\left(c-c_0\right)}$

Section D.2 Dynamic Beam equation

We get the dynamic beam equation by using the Euler-Lagrange equation for this action:

$$S=\int_0^L\left[\frac{1}{2}\mu\left(\frac{\partial w}{\partial t}\right)^2-\frac{1}{2}EI\left(\frac{\partial^2 w}{\partial x^2}\right)^2+q(x)w(x,t)\right]dx$$

Terms:

1. Kinetic energy per unit length: $\frac{1}{2}\mu\left(\frac{\partial w}{\partial t}\right)^2$
2. Potential energy due to internal forces: $\frac{1}{2}EI\left(\frac{\partial^2 w}{\partial x^2}\right)^2$
3. Potential energy due to the external load $q(x)$: $q(x)w(x,t)$

Use the Euler-Lagrange equation to find the function that minimizes the action *S*.

$$\boxed{\text{Answer: }\frac{\partial^2}{\partial x^2}\left(EI\frac{\partial^2 w}{\partial x^2}\right)=-\mu\frac{\partial^2 w}{\partial t^2}+q}$$

Section D.2.1 Lagrangian

The Lagrangian is the kinetic energy minus the potential energy: $L=T-V$.

Equations of motion come from the Euler-Lagrange equation: $\frac{\partial L}{\partial q}=\frac{d}{dt}\frac{\partial L}{\partial \dot{q}}$

(q is the function such that

$$q:[q,b]\in\mathbb{R}\to X$$
$$t\to x=q(t)$$

)

If there are two spatial dimensions, then this is a system, with one for each component:

$\frac{\partial L(t,q(t),q'(t))}{\partial x_i}-\frac{d}{dt}\frac{\partial L(t,q(t),q'(t))}{\partial v_i}=0$, for i = 1..n

Section D.2.1.1 e.g., particle in a conservative field:

$L(t,\mathbf{x},\mathbf{v}) = \frac{1}{2} m \sum_{i=1}^{3} v_i^2 - U(\mathbf{x})$, then we get:

$$\frac{\partial L(t,\mathbf{x},\mathbf{v})}{\partial x_i} = -\frac{\partial U(\mathbf{x})}{\partial x_i} = F_i(\mathbf{x})$$

and $\frac{\partial L(t,\mathbf{x},\mathbf{v})}{\partial v_i} = mv_i = p_i$

so that the Euler Lagrange equation is

$$\frac{\partial L}{\partial q} = \frac{d}{dt}\frac{\partial L}{\partial \dot{q}}$$

$$\rightarrow \left\{ F_i(\mathbf{x}) = \frac{d}{dt} p_i, \quad i = 1..3 \right.$$

Which is Newton's second law.

Now, since $\mathbf{v} = \dot{\mathbf{x}}$, this says: $\mathbf{F}(\mathbf{x}) = m\ddot{\mathbf{x}}$.

Section D.2.1.2 More complicated Lagrangians

One function of one variable, and only one derivative:

$$T - V = L(x, f, f_x)$$

$$\rightarrow \frac{\partial L}{\partial f} - \frac{d}{dx}\frac{\partial L}{\partial f'} = 0$$

(these should be fancy L's below, and

$$\{H\} = \{T\} + \{V\}$$

$$\{H\} = \sum_f \Pi_f \dot{f} - \{L\}$$

)

And without higher derivatives, $\{L\} = \{T\} - \{V\}$

(Where the $\{\ \}$ denote densities)

Extension #1: Higher derivatives:

$$T - V = \{L\}\left(x, f, f_x, f_{xx}, f_{xxx}, \cdots, \partial_x^n f\right)$$

$$\to \frac{\partial L}{\partial f} - \frac{d}{dx}\frac{\partial L}{\partial f_x} + \frac{d^2}{dx^2}\frac{\partial L}{\partial f_{xx}} - \cdots + (-1)^n \frac{d^n}{dx^n}\left(\frac{\partial L}{\partial\left(\partial_x^n f\right)}\right) = 0$$

(Longer equation)

/*

e.g., $T - V = x + 3\dot{x} + 7\ddot{x} + \dddot{x}$:

$$L = f\left(t, x, \dot{x}, \ddot{x}, \dddot{x}\right)$$

$$\to \frac{\partial L}{\partial x} - \frac{d}{dt}\frac{\partial L}{\partial \dot{x}} + \frac{d^2}{dt^2}\frac{\partial L}{\partial \ddot{x}} - \frac{d^3}{dt^3}\frac{\partial L}{\partial \dddot{x}} = 0$$

*/

Extension #2: More than one function:

$$T - V = L\left(x, f_1, f_2, \cdots, f_n, f_1', f_2', \cdots f_n'\right)$$

$$\to \left\{\frac{\partial L}{\partial f_i} - \frac{d}{dx}\frac{\partial L}{\partial\left(f_i'\right)} = 0, \qquad i = 1..n\right.$$

(Coupled equations)

e.g., object of mass 2 in free fall with some initial velocity:

$$T - V = \left(\dot{x}^2 + \dot{y}^2\right) + gy = L\left(t, x, \dot{x}, y, \dot{y}\right)$$

$$\to \begin{cases} \dfrac{\partial L}{\partial x} - \dfrac{d}{dt}\dfrac{\partial L}{\partial \dot{x}} = 0 \\ \dfrac{\partial L}{\partial y} - \dfrac{d}{dt}\dfrac{\partial L}{\partial \dot{y}} = 0 \end{cases}$$

gives: $\begin{cases} 0 - \dfrac{d}{dt}2\dot{x} = 0 \\ g - \dfrac{d}{dt}2\dot{y} = 0 \end{cases}$

Extension 3: Function of Several Variables

$$T - V = L\left(x_1, x_2, \cdots x_n, f, \partial_{x1} f, \partial_{x2} f, \cdots, \partial_{xn} f\right)$$

$$\rightarrow \frac{\partial L}{\partial f} - \sum_i \frac{d}{dx_i} \frac{\partial L}{\partial\left(\partial_{xi} f\right)} = 0$$

(Sum the derivatives)

Extension 3 & 1 together: Function of 2 variables with higher derivatives:

$$T - V = L\left(x, y, f, f_x, f_y, f_{xx}, f_{xy}, f_{yy}\right)$$

$$\frac{\partial L}{\partial f} - \frac{\partial}{\partial x}\frac{\partial L}{\partial f_x} - \frac{\partial}{\partial y}\frac{\partial L}{\partial f_y} + \frac{\partial^2}{\partial x^2}\frac{\partial L}{\partial f_{xx}} + \frac{\partial^2}{\partial x \partial y}\frac{\partial L}{\partial f_{xy}} + \frac{\partial^2}{\partial y^2}\frac{\partial L}{\partial f_{yy}} = 0$$

Section D.2.1.3 Derive the Dynamic Beam Equation

$$\{L\} = \{T\} - \{V\}$$

$$= \frac{1}{2}\mu\left(\frac{\partial w}{\partial t}\right)^2 - \frac{1}{2}EI\left(\frac{\partial^2 w}{\partial x^2}\right)^2 + qw = \{L\}\left(x, t, w, \dot{w}, w_x, w_{xx}\right)$$

$$= \frac{\mu}{2}\dot{w}^2 - \frac{EI}{2}w_{xx}^2 + qw$$

$$\text{Euler Lagrange} \rightarrow \frac{\partial\{L\}}{\partial w} - \frac{d}{dx}\left(\frac{\partial\{L\}}{\partial w_x}\right) - \frac{d}{dt}\frac{\partial\{L\}}{\partial \dot{w}} + \frac{d^2}{dx^2}\frac{\partial\{L\}}{\partial w_{xx}} = 0$$

$$q - \frac{d}{dx}(0) - \frac{d}{dt}(\mu\dot{w}) + \frac{d^2}{dx^2}\left(EIw_{xx}\right) = 0$$

$$q - \mu\ddot{w} + EIw_{xxxx} = 0$$

This differential equation admits the solutions with displacement:

$$\hat{w}(s) = A_1\left[\cosh\frac{\lambda}{L}s - \cos\frac{\lambda}{L}s + \frac{\cos\lambda + \cosh\lambda}{\sin\lambda + \sinh\lambda}\left(\sin\frac{\lambda}{L}s - \sinh\frac{\lambda}{L}s\right)\right]$$

and curvature:

$$\Delta c(s) \equiv \frac{\partial^2 \hat{w}}{\partial s^2} = A_1\frac{\lambda^2}{L^2}\left[\cosh\frac{\lambda}{L}s + \cos\frac{\lambda}{L}s + \frac{\cos\lambda + \cosh\lambda}{\sin\lambda + \sinh\lambda}\left(-\sin\frac{\lambda}{L}s - \sinh\frac{\lambda}{L}s\right)\right]$$

where λ are the solutions to the equation $\cosh(\lambda)\cos(\lambda)+1=0$. For the first mode, $\lambda \approx 1.875$.

Appendix E MATLAB program to explore Multimorph Stress

```
clear all;
close all;

t=[2/3, 1/3]*48.8e-6;  %Layer thickness; layer 1, layer 2
E=[111.4e9, 86.7e9];  %Layer young's modulus
rho=[2300,2600]; %Layer density
rhom=sum(rho.*t)/sum(t); %weighted average beam density
L=9.5e-3; %Beam length
B=L/4; %Beam width
A=t*B;  %Layer Cross-sectional area
epfree=[4e-6, 16e-6]*(-621.1485354);


nL=length(t); %number of layers


y=0;
for(i=1:length(t))
    y(i)=sum(t(1:(i-1))) + t(i)/2; %yi is the coordinate of the centroid of the ith layer
end
Ybar=sum(A.*E.*y)/sum(A.*E);
I0=B*t.^3/12;
EI=sum(E.*I0 + A.*E.*(y-Ybar).^2);
fres=1.875^2/L^2/(2*pi)*sqrt(EI/B/sum(t)/rhom);
%DeVoe formulation; from the DeVoe Paper
Ddv=1/sum(E.*I0)*y
Adv=zeros(nL);
for(i=1:nL-1)
    %Generate the 2 nonzero entries in Row i:
    Adv(i,i) = 1/(A(i)*E(i));
    Adv(i,i+1) = -1/(A(i+1)*E(i+1));
end
for(i=1:nL)
    %Generate the row of 1s in the last row of A:
    Adv(nL,i)=1;
end
Bdv=[t(1:nL-1)+t(2:nL),0]';
Cdv=[epfree(2:nL)-epfree(1:nL-1), 0]';
cdv=Ddv*Adv^-1*Cdv/(1-0.5*Ddv*Adv^-1*Bdv);
phiDegdv=cdv*L*180/pi


Fdv=Adv^-1*(1/2*cdv*Bdv+Cdv);
sigmaCompressive=Fdv'./A-(E.*t)/2*cdv;
sigmaTensile=Fdv'./A+(E.*t)/2*cdv;
```

```
epCompressive=sigmaCompressive./E
epTensile=sigmaTensile./E

%% Formulation 1 for res frequency (only works for 2 layers)
m=t(1)/t(2);
n=E(1)/E(2);
Ey=m*E(1)/((m+1)^3*(m*n+1))*(4+6*m+4*m^2+n*m^3+1/(n*m));
I=B*(sum(t))^3/12;
EI1=Ey*I;
fres1=1.875^2/L^2/(2*pi)*sqrt(EI1/B/sum(t)/rhom)
%%Timoshenko Formulation for curvature:
%c = (strain difference)/h * curvrel, and:

DelEp=(epfree(2)-epfree(1));
h=sum(t);
cTimoshenko=6*DelEp*(1+m)^2/((m^2+1/n/m)*(1+n*m)+3*(1+m)^2)/h;
phiDegTimoshenko=cTimoshenko*L*180/pi;
```

Appendix F Deriving Beam Displacement due to Curvature

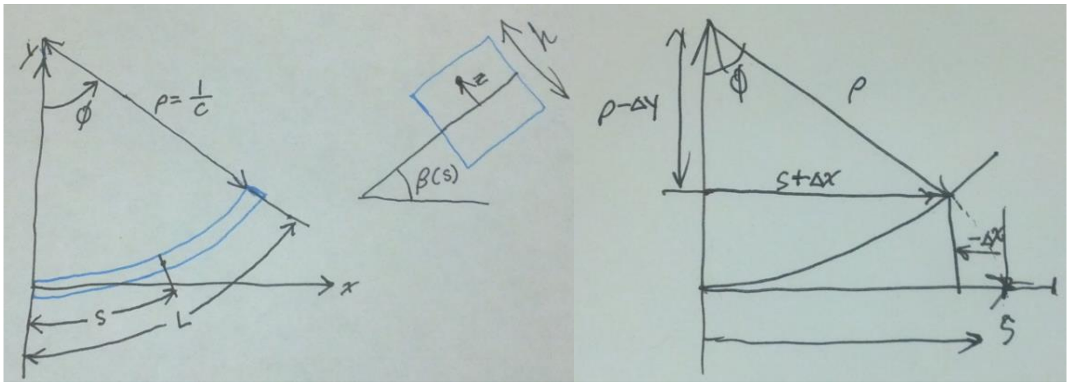
Section F.1 Displacement of a point on the neutral axis (z = 0)

It's easiest to first find the displacement of points on the *neutral axis* of the cantilever: that axis which experiences no longitudinal strain. For a *symmetric* bimorph cantilever actuator, the neutral axis is in the vertical middle: $z=0$. *(The symmetric case is the one assumed here for simplicity, but this leads to no loss of generality, for if the actuator is not symmetric, we will still be able to use the resulting formulas if we adjust the cantilever extent. E.g., for a neutral axis at the absolute bottom, instead define the cantilever extent as* $(s,z)=(0..L,0..h)$*.)*

The arc length *s* of the neutral axis is related to ϕ & ρ through $s=\rho\times\phi(s,c)$, so that

$$\boxed{\phi(s,c)=\frac{1}{\rho}s=cs}$$

N.B.: Even off the axis, ϕ is not a function of *z*, as we assume that plane cross sections remain plane in the cantilever.



Appendix Figure F-1: Variables for beam displacement derivation

To find the external displacements of each internal point, use Appendix Figure F-1 to write:

$$\sin(\phi(s,c))=\frac{s-(-\Delta x(s,0,c))}{\rho},\qquad \cos(\phi(s,c))=\frac{\rho-\Delta y(s,0,c)}{\rho}$$

Which lead to:

$$\sin(\phi) = \frac{s+\Delta x}{\rho} = \frac{s+\Delta x}{1/c}$$

$$\rightarrow \frac{1}{c}\sin(sc) = s + \Delta x$$

$$\therefore \Delta x = \frac{1}{c}\sin sc - s = \frac{1}{c}\left(\sin(sc) - sc\right)$$

and

$$\cos(\phi) = \frac{\rho - \Delta y}{\rho} = \frac{\frac{1}{c} - \Delta y}{1/c}$$

$$\rightarrow \frac{1}{c}\cos(sc) = \frac{1}{c} - \Delta y$$

$$\therefore \Delta y = \frac{1}{c}\left(1 - \cos(sc)\right)$$

$$\boxed{\Delta\mathbf{r}(s,0,c) = \frac{1}{c}\begin{bmatrix}\sin(sc) - sc \\ 1 - \cos(sc)\end{bmatrix}}$$

In terms of ϕ, $\Delta\mathbf{r}(s,0,\phi) = \frac{s}{\phi}\begin{bmatrix}\sin\phi - \phi \\ 1 - \cos\phi\end{bmatrix}$, which is more useful at times.

Given that the initial external coordinate from internal coordinate $s,0$ is $\mathbf{r}(s,0,0) = \begin{bmatrix} s \\ 0 \end{bmatrix}$, the new coordinate is

$$\mathbf{r}(s,0,c) = \mathbf{r}(s,0,0) + \Delta\mathbf{r}(s,0,c) = \frac{1}{c}\begin{bmatrix}\sin(sc) \\ 1 - \cos(sc)\end{bmatrix}$$

Section F.2 Displacement of a point at arbitrary z:

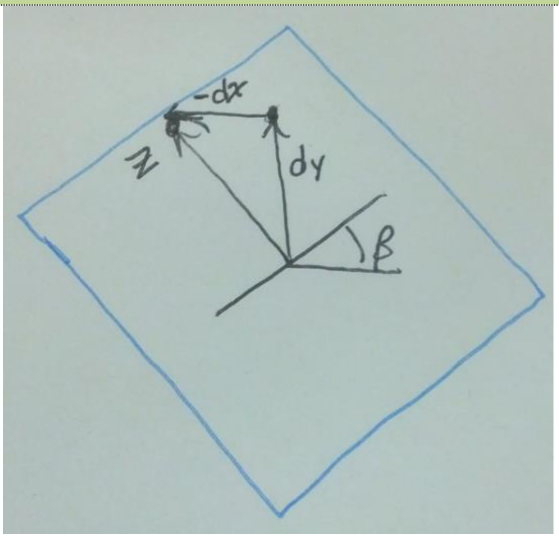


Figure F-2: Off-axis displacement due to bending angle of a beam section

Referring to Figure F-2, the off-axis point originally at (s,z) will find itself moving to a new position of $\mathbf{r}(s,z,c)=\mathbf{r}(s,z,0)+\Delta\mathbf{r}(s,0,c)+\begin{bmatrix}-z\sin\beta\\+z\cos\beta\end{bmatrix}$

β, the angle that the section at s's neutral axis makes with the horizontal, is related to the curvature and arclength via: $\beta=sc$ (for pure cylindrical bending, $\beta(s)=\phi(s)$. Later we'll examine resonant bending where this is not the case).

Given that the original position was $\mathbf{r}(s,z,0)=\begin{bmatrix}s\\z\end{bmatrix}$ and the neutral-axis displacement is $\Delta\mathbf{r}(s,0,c)=\frac{1}{c}\begin{bmatrix}\sin sc-sc\\1-\cos sc\end{bmatrix}$ (as given above), this makes the total displacement:

$$\Delta\mathbf{r}(s,z,c)=\frac{1}{c}\begin{bmatrix}\sin sc-sc\\1-\cos sc\end{bmatrix}+z\begin{bmatrix}-\sin sc\\\cos sc\end{bmatrix}$$

and the new coordinates of the point are: $\mathbf{r}=\begin{bmatrix}s\\z\end{bmatrix}+\frac{1}{c}\begin{bmatrix}\sin sc-sc\\1-\cos sc\end{bmatrix}+z\begin{bmatrix}-\sin sc\\\cos sc\end{bmatrix}$

Appendix G Chain rule to simplify the RS definition:

Unfortunately, the functional form for β is not easy to analytically integrate to find the displaced position, so most of the work dealing with this is going to be numeric. We can however numerically simplify the z-correction term using the chain rule:

$$\mathbf{r}(s,z,t)=\begin{bmatrix}0\\z\end{bmatrix}+\int_0^s\left(1+\varepsilon_0(s,t)-z\frac{\partial\beta(s,t)}{\partial s}\right)\begin{bmatrix}\cos(\beta(s,t))\\\sin(\beta(s,t))\end{bmatrix}ds$$

$$=\begin{bmatrix}0\\z\end{bmatrix}+\int_0^s\left(1-z\frac{\partial\beta(s,t)}{\partial s}\right)\begin{bmatrix}\cos(\beta(s,t))\\\sin(\beta(s,t))\end{bmatrix}ds$$

$$=\begin{bmatrix}0\\z\end{bmatrix}+\int_0^s\begin{bmatrix}\cos\beta\\\sin\beta\end{bmatrix}ds-z\int_0^s\frac{\partial\beta}{\partial s}\begin{bmatrix}\cos\beta\\\sin\beta\end{bmatrix}ds$$

$$=\begin{bmatrix}0\\z\end{bmatrix}+\int_0^s\begin{bmatrix}\cos\beta\\\sin\beta\end{bmatrix}ds-z\int_0^s\frac{\partial\beta}{\partial s}\begin{bmatrix}\cos\beta\\\sin\beta\end{bmatrix}ds$$

Examine the integral using the chain rule:

$$\int_0^s\frac{\partial\beta}{\partial s}\begin{bmatrix}\cos\beta\\\sin\beta\end{bmatrix}ds=\int_0^\beta\begin{bmatrix}\cos\beta\\\sin\beta\end{bmatrix}d\beta$$

$$=\begin{bmatrix}\sin\beta\\-\cos\beta\end{bmatrix}-\begin{bmatrix}0\\-1\end{bmatrix}=\begin{bmatrix}\sin\beta\\1-\cos\beta\end{bmatrix}$$

Therefore, you could more compactly write the position as:

$$\mathbf{r}=\begin{bmatrix}0\\z\end{bmatrix}-z\begin{bmatrix}\sin\beta\\1-\cos\beta\end{bmatrix}+\int_0^s\begin{bmatrix}\cos\beta\\\sin\beta\end{bmatrix}ds$$

$$=\int_0^s\begin{bmatrix}\cos\beta\\\sin\beta\end{bmatrix}ds+z\begin{bmatrix}-\sin\beta\\\cos\beta\end{bmatrix}$$

(The z-correction working the same as for cylindrical bending).

And so, alternatively $\mathbf{r}=\int_0^s\begin{bmatrix}\cos\beta\\\sin\beta\end{bmatrix}ds+z\begin{bmatrix}-\sin\beta\\\cos\beta\end{bmatrix}$, although this is somewhat less convenient when using curvature instead of β.

Appendix H COMSOL Simulations Exploring 3d Boundary Condition Corrections to Young's Modulus and Piezoelectric Coupling Coefficients

DeVoe and Pisano mention in [28] that for fixed cantilever boundary conditions, the effective Youngs modulus of a bending beam is $E' = \frac{E}{1-\nu^2}$, and the effective piezoelectric coefficient is $d_{31}' = d_{31}(1+\nu)$. To explore whether these are the case for III-V materials and Quartz, I carried out a number of simulations in COMSOL Multiphysics 3.5a.

Section H.1 InGaP_2dVolt

Built a 2d rectangular grid and explored the effective d_{31} required so that the strain would develop using $\varepsilon_1 = d_{31}E_3 = d_{31}\frac{\Delta V}{t}$ in response to voltage over layer thickness *t*, for a III-V material using InGaP's d_{31} of 5.791e-13.

From [66], III-V materials have the piezoelectric coupling matrix

$$\mathbf{d}(\phi) = \frac{d_{14}}{2}\begin{bmatrix} 0 & 0 & 0 & 2a_2 & -2b_2 & 0 \\ 0 & 0 & 0 & 2b_2 & 2a_2 & 0 \\ -b_2 & b_2 & 0 & 0 & 0 & 2a_2 \end{bmatrix}$$

where the *z* axis is $\langle 100 \rangle$, the *x* axis makes an angle of ϕ with $\langle 0\bar{1}1 \rangle$, and $a_2 = \sin 2\phi, b_2 = \cos 2\phi$. The maximum linear piezoelectric effect therefore couples electric fields in the $\langle 100 \rangle$ direction to strains in the $\left[0\bar{1}1\right]$ ones, and $d_{31} = -d_{32}$: applied voltage in z extends the material in x and shrinks it in y. This attempted shrinking in *y* is resisted by the cantilever boundary conditions, and so actually results in a similar effect to *extending* the beam in the y-direction with strain: it should shrink in the other directions by the attempted strain multiplied by the poisson's ratio.

Result: Longitudinal strain and thus effective d_{31} was 66.4% of the free strain; that's the same as if one were to scale $d_{31} \rightarrow d_{31}(1-\nu)$ (ν = 0.33).

To ensure this understanding is correct, I carried out a number of additional tests:

1) **Remove the d_{32} component.** The effective strain should now be the one predicted using the actual d_{31}, since the material no longer attempts to shrink in *y*. Verified.
2) **Reverse the d_{32} component.** Effective strain should now be 33% *larger*, as the material attempts to *extend* in *y*. Verified.
3) **Altered the Poisson ratio to 0.2.** material should now develop 80% of the free strain. Verified.

Result: these COMSOL simulations confirmed that DeVoe's statement that for plane strain conditions $d_{effective} = d_{31}(1+\nu)$ applies to "typical isotropic-E piezoelectric" materials with $d_{13} = d_{23}$. For a III-V material, E is still isotropic but $d_{13} = -d_{23}$ and using COMSOL we found that $d_{effective} = d_{31}(1-\nu)$.

Eigenfrequency and bending simulations demonstrate similarly that:

1. E_Y effective for quartz (of the orientation I'm using; x-thickness, y-length, z-width of a cantilever) (for resonance agreement with COMSOL over a large range) is actually 86.37298814 GPa (not actually that many sig figs)
   a. This particular E_Y was defined as the one that gives identical theory and COMSOL results for the resonant frequency of a 30μm Thick 1mm long quartz-quartz cantilever with ground plates between them and on the top & bottom:
   b. This works the same for bending as Eigenfrequency Verified
2. This is modified by the boundary conditions due to piezoelectric effects, and COMSOL reflects this in a sensible way.

Note: At least for version 3.5a, COMSOL has problems modelling h <=30μm unless L <=8mm. It has less problems with h<=120μm, L<=64mm, even though L/h is quite a bit larger and in fact so is L^2/h. These problems are also present for simple isotropic materials.

Appendix I RS Mode Energy Balance

Section I.1 In general

The resonant frequency of the first mode of a straight cantilever is:

$$f_{resStraightBeam} = \frac{\lambda^2}{L^2 2\pi}\sqrt{\frac{EI}{Bh\rho_m}}$$

where L is cantilever length, B is the width, h is the thickness, ρ_m is the density, EI is flexural rigidity, and λ is the first root of to $\cos\lambda\cosh\lambda+1=0$.

Preview:

For a curved cantilever, I've found that this formula still works if you apply a correction factor that depends on the natural bending angle, $f_{resCor}\left(\phi_0\right)$:

$$f_{res} = \frac{\lambda^2}{L^2 2\pi}\sqrt{\frac{EI}{Bh\rho_m}}f_{resCor}$$

where $f_{resCor}\left(\phi\right)=1+0.0202\phi^2+0.000242\phi^4$, where ϕ is in radians. This results from truncating a Taylor expansion and is verified with ANSYS mechanical simulations, but is also analytically derived by equating the kinetic and potential energy for the RS mode shape. The following sections do just that.

Section I.2 Energy Derivations

With the RS mode asserted, we can calculate the kinetic and potential energy of the beam. At resonance, the kinetic and potential energy must balance, so by setting these quantities equal, we can derive the resonant frequency of the beam as a function of the static bending angle. Then, we can compare this to numerical results for the resonant frequency as a test of the RS mode.

Section I.2.1 Potential Energy Derivation

Summary of this section: For RS bending, the extra potential energy from bending is $\boxed{\Delta U = \tfrac{1}{2} EI \int_0^L \Delta c^2 ds = \frac{1}{2} EI \left(\frac{\lambda^4}{4} \frac{A_r^2}{L} \right)}$, where $\lambda = 1.8751$ is the first root of $\cos\lambda \cosh\lambda + 1 = 0$.

To derive this potential energy formula:

1. Prove that we can work with Δc rather than the absolute curvature.
2. Write an expression for the potential energy in terms of this Δc
3. Substitute and nondimensionalize
4. Numerically evaluate the integral using the assumed form of Δc

Section I.2.1.1 Proof that Δc replaces c for strain energy for bending from c0:

Neglecting shear strain, $U = \int dU = \int_{Beam} \tfrac{1}{2} E\varepsilon^2 dV$

What if the beam has a resting strain distribution which gives rise to a static curvature?

$$\varepsilon_{mech} = \varepsilon - \varepsilon_{free}$$

$$\varepsilon_{mech} = \varepsilon_0 + \varepsilon_{bending} - \varepsilon_{free}$$

$$\frac{\sigma}{E} = \varepsilon_0 - zc - \varepsilon_{free}$$

(where ε_{free} is the strain that would occur if the layer were released from the composite beam, and ε_0 is the strain at $z = 0$ that is actually manifested.)

The resting curvature is determined from the beam's equilibrium conditions of no net force or moment on any cross-section:

1. Net force on any cross section is 0: $\int_A \sigma dA = \int_A E\left(\varepsilon_0 - zc_0 - \varepsilon_{free}\right) dA = 0$
2. Net moment on any cross section is 0: $\int_A \sigma z dA = \int_A E\left(\varepsilon_0 - zc_0 - \varepsilon_{free}\right) z dA = 0$
   a. → $\int_A E\left(\varepsilon_0 - \varepsilon_{free}\right) z dA = EIc_0$.

The relevant strain for the potential energy is this "mechanical strain", since that's the strain that produces a stress:

$$U=\int dU=\int_{Beam}\tfrac{1}{2}E\left(\varepsilon_0-zc-\varepsilon_{free}\right)^2 dV$$

Defining $\varepsilon_{BuiltIn}\equiv\varepsilon_0-\varepsilon_{free}$ as the resting non-bending strain in the curved beam, the true change in potential energy due to a change in curvature alone (i.e., change in longitudinal strain at z = 0 is 0) is:

$$\begin{aligned}
\Delta U &= U-U_0\\
&=\int_{beam}\tfrac{1}{2}E\left(\varepsilon_{BuiltIn}-zc\right)^2 dV-\int_{beam}\tfrac{1}{2}E\left(\varepsilon_{BuiltIn}-zc_0\right)^2 dV\\
&=\int_{beam}\tfrac{1}{2}E\begin{bmatrix}\varepsilon_{BuiltIn}^2-2\varepsilon_{BuiltIn}zc+z^2c^2\\ -\left(\varepsilon_{BuiltIn}^2-2\varepsilon_{BuiltIn}zc_0+z^2c_0^2\right)\end{bmatrix}dV\\
&=\int_{beam}\tfrac{1}{2}E\left(-2\varepsilon_{BuiltIn}z\left(c-c_0\right)+z^2\left(c^2-c_0^2\right)\right)dV\\
&=\int_{beam}\tfrac{1}{2}E\left(-2\varepsilon_{BuiltIn}z\left(\left(c_0+\Delta c\right)-c_0\right)+z^2\left(\left(c_0+\Delta c\right)^2-c_0^2\right)\right)dV\\
&=\int_{beam}\tfrac{1}{2}E\left(-2\varepsilon_{BuiltIn}z\Delta c+z^2\left(\Delta c^2+2c_0\Delta c\right)\right)dV\\
&=\int_{beam}\tfrac{1}{2}E\left(-2\varepsilon_{BuiltIn}z\Delta c\right)dV+\int_{beam}\tfrac{1}{2}E\left(z^2\left(\Delta c^2\right)\right)dV+\int_{beam}\tfrac{1}{2}E\left(z^2\left(2c_0\Delta c\right)\right)dV
\end{aligned}$$

Evaluating each integral separately:

1)

$$\begin{aligned}
\int\tfrac{1}{2}E\left(-2\varepsilon_{builtin}z\Delta c\right)dV &= \tfrac{1}{2}\int_0^L\Delta c ds\int_A E\left(-2\varepsilon_{builtin}z\right)dA\\
&=\tfrac{1}{2}\int_0^L\Delta c ds\int_A\left(-2\sigma_{builtIn}z\right)dA
\end{aligned}$$

and $\int_A E\left(\varepsilon_{builtin}z\right)dA=\sum M_{builtin}=EIc_0$, as above for a total term of: $-\tfrac{1}{2}EIc_0\int_0^L 2\Delta c ds$

2)

$$\int_{beam} \tfrac{1}{2} Ez^2 \Delta c^2 dV = \tfrac{1}{2} EI \int_0^L \Delta c^2 ds$$

(This is found to be $\tfrac{1}{2} EI \int_0^L \Delta c^2 ds = \frac{1}{2} EI \left(3.0906 \frac{A_r^2}{L} \right)$ *in the next section)*

3)

$$\int_{beam} \tfrac{1}{2} Ez^2 2c_0 \Delta c dV = \tfrac{1}{2} EIc_0 \int_0^L 2\Delta c ds$$

(cancels with term #1)

Therefore, without an additional axial strain component to the resonance, the stored energy is based only on the difference in curvature from equilibrium.

Section I.2.1.2 Calculating RS Potential energy for Δc

In the previous section, we found that the extra potential energy beyond neutral curvature for the beam is:

$$\Delta U = U - U_0 = \int_{beam} \tfrac{1}{2} Ez^2 \Delta c^2 dV = \tfrac{1}{2} EI \int_0^L \Delta c^2 ds$$

Where $\Delta c = \frac{\partial \Delta\beta}{\partial s}$, $\Delta\beta \equiv A_r \frac{\lambda}{2} \left[\sinh \frac{\lambda s}{L} + \sin \frac{\lambda s}{L} + \frac{\cos\lambda + \cosh\lambda}{\sin\lambda + \sinh\lambda} \left(\cos \frac{\lambda}{L} s - \cosh \frac{\lambda}{L} s \right) \right]$,

and $EI \equiv \int_{CrossSection} Ez^2 B dz = \sum_i (EI)_i$

We can numerically solve for λ and then evaluate this ΔU integral for the Δc of RS bending to find. This is carried out in the next subsections, and finds that:

$$\lambda = 1.87510406871196...$$

$$\Delta U = \frac{1}{2} EI \frac{3.09059 A_r^2}{L}$$

and also that

$$\Delta U = \frac{1}{2} EI \frac{\lambda^4}{4} \frac{A_r^2}{L} \qquad (7.1)$$

I.2.1.2.1 Numeric evaluation for lambda

λ is defined as the solution to $\frac{\cos\lambda+\cosh\lambda}{\sin\lambda+\sinh\lambda}=1$; resulting from setting no bending moment or shear stress at the tip:

$$\left.\Delta c\right]_{s=L}=\left.\frac{\partial}{\partial s}\Delta c\right]_{s=L}=0:$$

Apply the "no bending moment at the tip" boundary condition:

$$\left.\Delta c\right]_{s=L}=A_1\frac{\lambda^2}{L^2}\left[\cosh\lambda+\cos\lambda+\frac{\cos\lambda+\cosh\lambda}{\sin\lambda+\sinh\lambda}\left(-\sin\lambda-\sinh\lambda\right)\right]$$
$$=0$$

(by definition)

Apply the "no shear stress at the tip" boundary condition:

$$\left.\frac{\partial}{\partial s}\Delta c\right]_{s=L}=0$$
$$A_1\frac{\lambda^3}{L^3}\left[\sinh\lambda-\sin\lambda+\frac{\cos\lambda+\cosh\lambda}{\sin\lambda+\sinh\lambda}\left(-\cos\lambda-\cosh\lambda\right)\right]=0$$
$$\frac{\left(\cos\lambda+\cosh\lambda\right)^2}{\sin\lambda+\sinh\lambda}=-\sin\lambda+\sinh\lambda$$
$$\cos^2\lambda+2\cos\lambda\cosh\lambda+\cosh^2\lambda=-\sin^2\lambda+\sinh^2\lambda$$
$$1+2\cos\lambda\cosh\lambda+1=0$$

(using the fundamental trig and hyperbolic identities, $\sin^2 x+\cos^2 x=1$ and $\cosh^2 x-\sinh^2 x=1$)

$$\cos\lambda\cosh\lambda=-1$$

This can be evaluated numerically; the function `calc_symbolic_delta_U()`: from `RSVelocityWork.py` comes up with

```
lamSoln = 1.87510406871196
```

I.2.1.2.2 Numeric evaluation of ΔU

$$\Delta c = \frac{\partial \Delta\beta}{\partial s}$$

$$= A_r \frac{\lambda^2}{2L}\left[\cosh\frac{\lambda s}{L} + \cos\frac{\lambda s}{L} + \frac{\cos\lambda + \cosh\lambda}{\sin\lambda + \sinh\lambda}\left(-\sin\frac{\lambda}{L}s - \sinh\frac{\lambda}{L}s\right)\right]$$

i.e, $\Delta c = \frac{A_r}{L} f\left(\frac{s}{L}\right)$

So,

$$\Delta U = \tfrac{1}{2} EI \int_0^L \Delta c^2 ds$$

$$S \equiv \frac{s}{L},\ dS = \frac{ds}{L}$$

$$\Delta U = \tfrac{1}{2} EI \int_0^1 \left(\frac{A_r}{L} f(S)\right)^2 (LdS)$$

$$\Delta U = \tfrac{1}{2} EI \frac{A_r^2}{L} \int_0^1 \left(f(S)\right)^2 dS$$

Where $f(S) \equiv \frac{\lambda^2}{2}\left[\cosh(\lambda S) + \cos(\lambda S) + \frac{\cos\lambda + \cosh\lambda}{\sin\lambda + \sinh\lambda}(-\sin\lambda S - \sinh\lambda S)\right]$

(Now the integral has been nondimensionalized.)

Using the precise `lamSoln`, find using `calc_symbolic_delta_U()` that

$$\int_0^1 f^2(S)\, dS =$$

3.0905894240098 (1000-pt midpoint rule)

3.09059048756361 (2000-pt midpoint rule)

3.09059068451803 (3000-pt midpoint rule)

So that $\boxed{\Delta U = \tfrac{1}{2} EI \frac{A_r^2}{L} 3.09059 = \tfrac{1}{2} EI \frac{A_r^2}{L}\frac{\lambda^4}{4}}$ (Using `lamSoln = 1.87510406871196` with a 3000-point midpoint rule)

As expected, the max potential energy occurs when $A_r = A_{r0}\sin\omega t$ is its max value of A_{r0}, and the potential energy scales with the flexural rigidity and the square of the amplitude (as in a Hooke's law spring).

Section I.2.2 Kinetic energy

$$K = \int_{wing} \tfrac{1}{2} v^2 dm$$

$$K = \tfrac{1}{2}\int_0^L v^2\left(s,t\right)\mu_m ds$$

(μ_m is linear mass density.)

Unfortunately, this integral is very difficult to do for the RS mode. The difficulty lies in the iterative position definition:

$$\mathbf{r} = \int_0^s \begin{bmatrix} \cos\beta \\ \sin\beta \end{bmatrix} ds + z\begin{bmatrix} -\sin\beta \\ \cos\beta \end{bmatrix}$$

where $\beta = \beta_0 + A_r\dfrac{\lambda}{2}\left[\sinh\dfrac{\lambda s}{L} + \sin\dfrac{\lambda s}{L} + \dfrac{\cos\lambda + \cosh\lambda}{\sin\lambda + \sinh\lambda}\left(\cos\dfrac{\lambda}{L}s - \cosh\dfrac{\lambda}{L}s\right)\right]$ and $A_r = A_{r0}\sin\omega t$, making the velocity equal to:

$$\mathbf{v}\left(s,z,t\right) \equiv \frac{\partial\mathbf{r}\left(s,z,t\right)}{\partial t} = \frac{\partial}{\partial t}\int_0^s \begin{bmatrix} \cos\beta \\ \sin\beta \end{bmatrix} ds + \frac{\partial}{\partial t} z\begin{bmatrix} -\sin\beta \\ \cos\beta \end{bmatrix}$$

Since z is very small, neglect the modification to velocity from off-axis components:

$$\mathbf{v} \approx \frac{\partial}{\partial t}\int_0^s \begin{bmatrix} \cos\beta \\ \sin\beta \end{bmatrix} ds$$

$$\frac{\partial}{\partial t}\beta = \frac{\partial}{\partial t}\left(\beta_0 + \Delta\beta\right)$$

$$= 0 + \frac{\partial \Delta\beta}{\partial t}$$

$$= \left(\omega A_{r0}\cos\omega t\right)\frac{\lambda}{2}\left[\sinh\frac{\lambda s}{L} + \sin\frac{\lambda s}{L} + \frac{\cos\lambda + \cosh\lambda}{\sin\lambda + \sinh\lambda}\left(\cos\frac{\lambda}{L}s - \cosh\frac{\lambda}{L}s\right)\right]$$

$$= \omega\frac{\cos\omega t}{\sin\omega t}\Delta\beta$$

So

$$\mathbf{v} \approx \omega\frac{\cos\omega t}{\sin\omega t}\int_0^s \Delta\beta\begin{bmatrix}\cos\beta\\ \sin\beta\end{bmatrix}ds$$

And even now, this is impossible to do analytically.

But we *can* do it numerically.

Assert that the maximum velocity should be at the equilibrium crossing point.

To find the velocity numerically:

1. Find the position of the wing at t = t_0
2. Find the position of the wing at t = t_0+Δt
3. Find the displacement by subtracting these two
4. Find the velocity by dividing this by Δt.

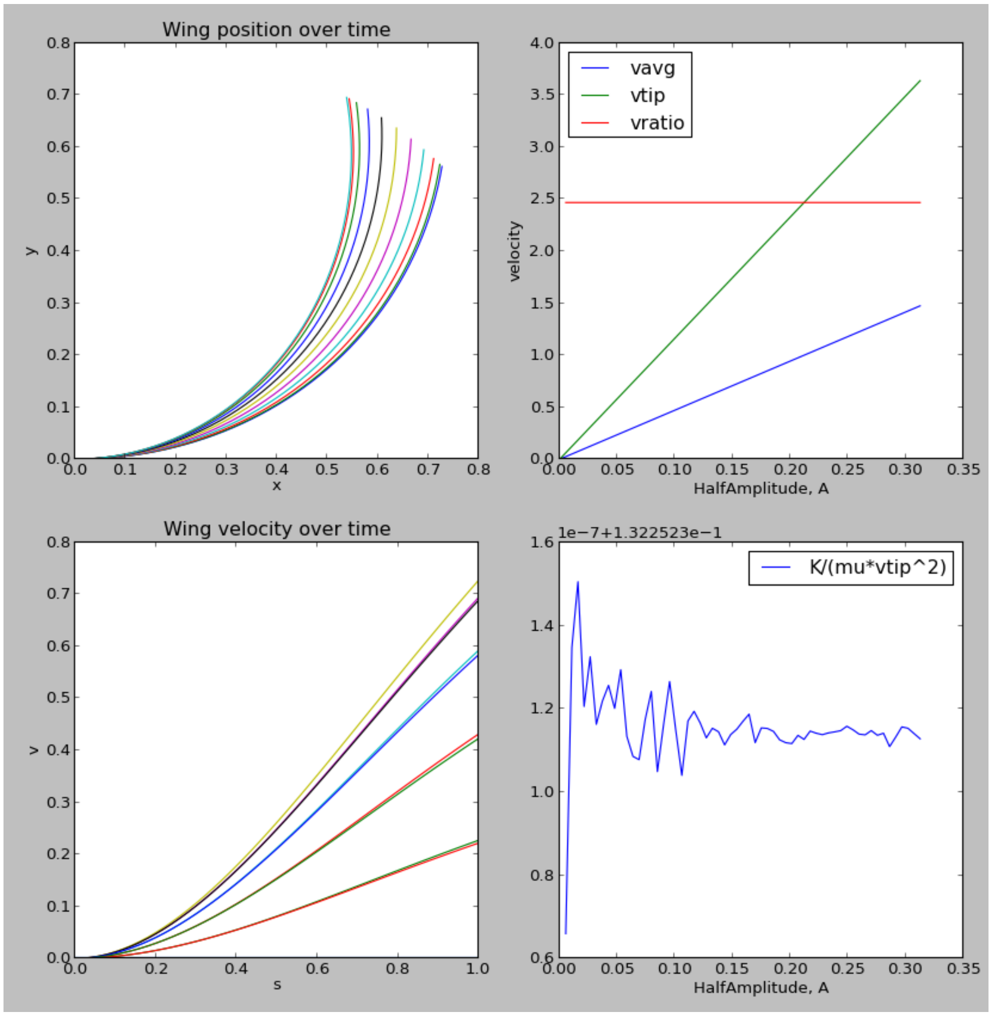
Results: Analyzing K vs. L, ω, Ar, and ϕ using the program in Appendix J:

$$\boxed{K_{\max} = \frac{A_{r0}^2\omega^2 L^3}{8 f_{resCor}^2}\left(\rho_m Bh\right)}, \text{ where } f_{resCor} \equiv 1 + 2.0239\text{e-}2\phi_0^2 + 2.4241\text{e-}4\phi_0^4$$

Section I.2.2.1 Kinetic energy scaling

In this section, we plot the wing velocity (tip and average velocity, as well as the ratio between the two), and the quantity $\frac{K}{\mu_m v_{tip}^2}$ to investigate the peak kinetic energy and velocity scaling for the RS mode vs. several parameters, amplitude *A*, wing length *L*, and frequency *ω*.

I.2.2.1.1 K vs. A, fixed L



Appendix Figure I-1: Kinetic energy divided by $\mu_m v_{tip}^2$ is independent of amplitude for fixed wing length, static bending angle, and frequency

At phi0=90.0,L=1,A=0.0625,omega=6.28318530718,

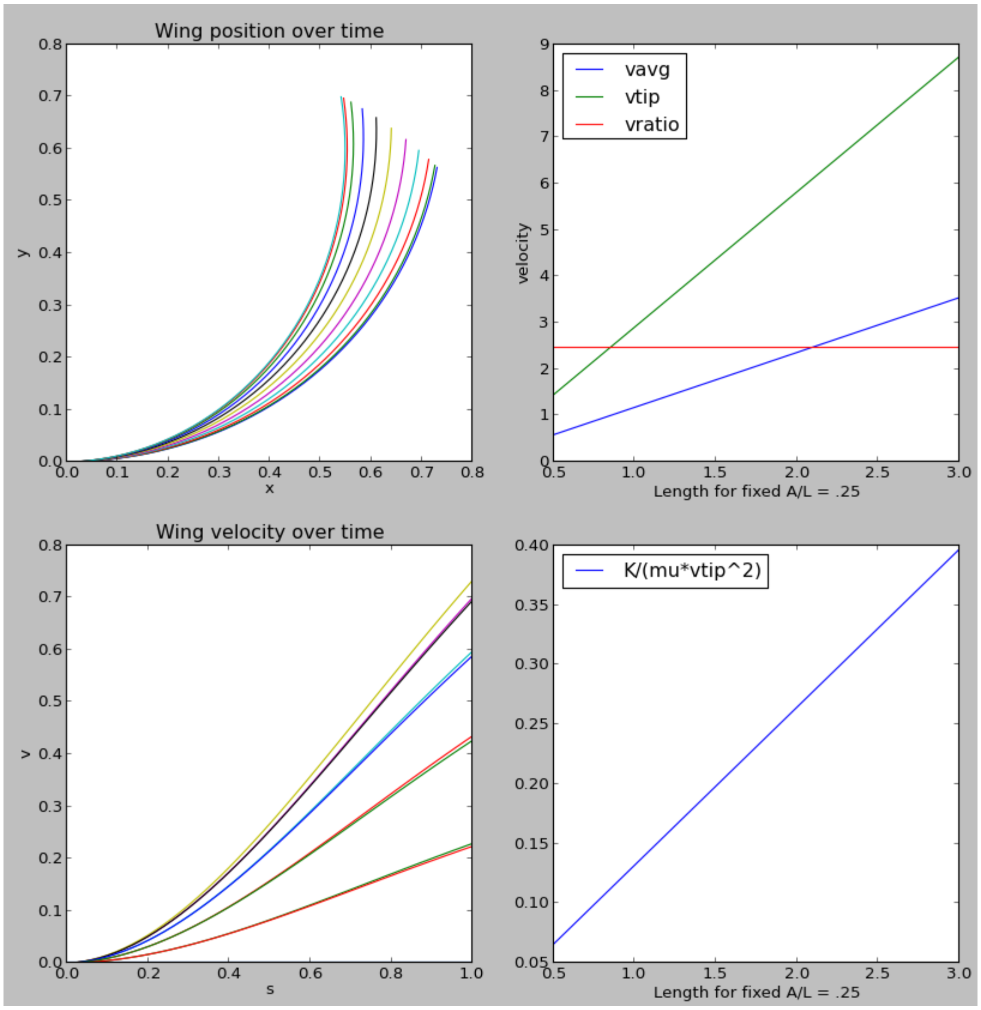
$$\frac{K}{\mu_m v_{tip}^2} = 0.132$$; and this is:

independent of A

I.2.2.1.2 K vs. L

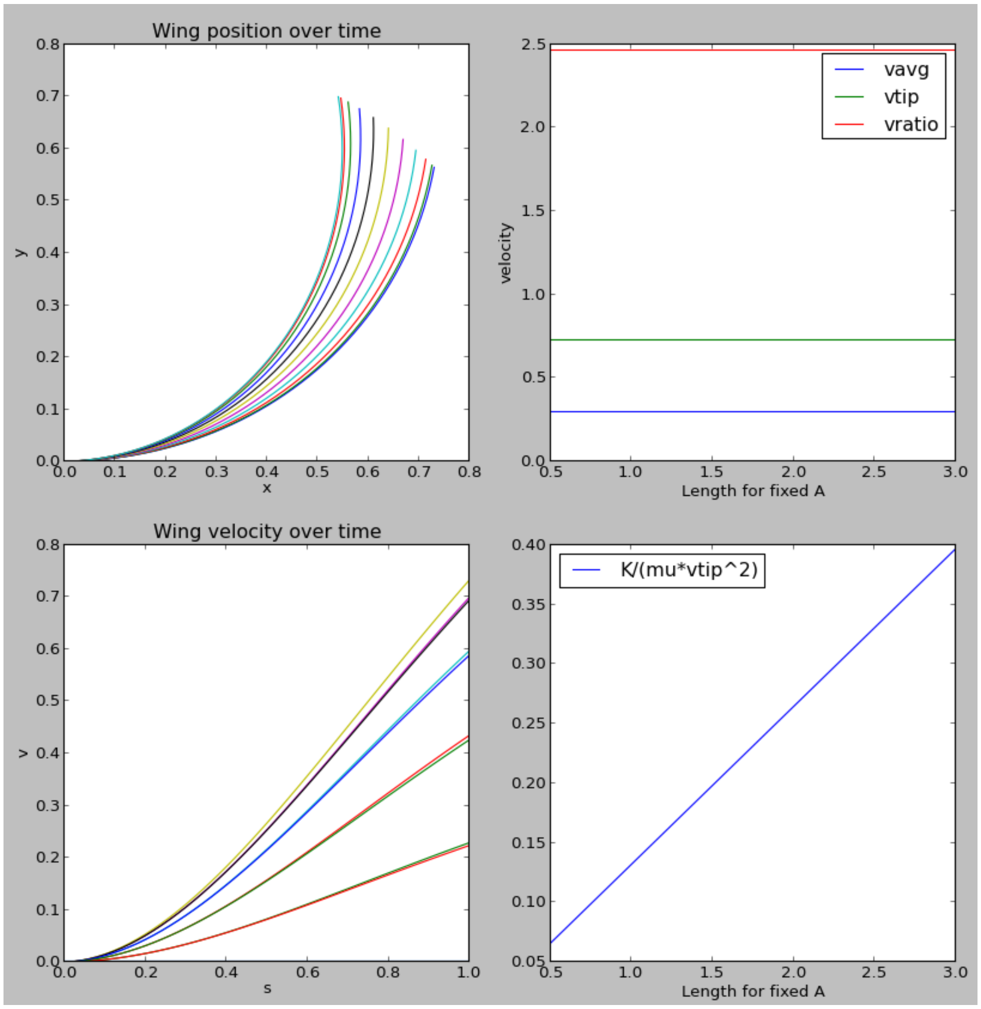
For both fixed A/L and variable A/L, (K/μ-vtip^2) is directly proportional to L and independent of A.

I.2.2.1.3 fixed A/L



Appendix Figure I-2: Kinetic energy divided by $\mu_m v_{tip}^2$ is linear with length for fixed amplitude per length, static bending angle, and frequency

I.2.2.1.4 K vs. L, fixed A

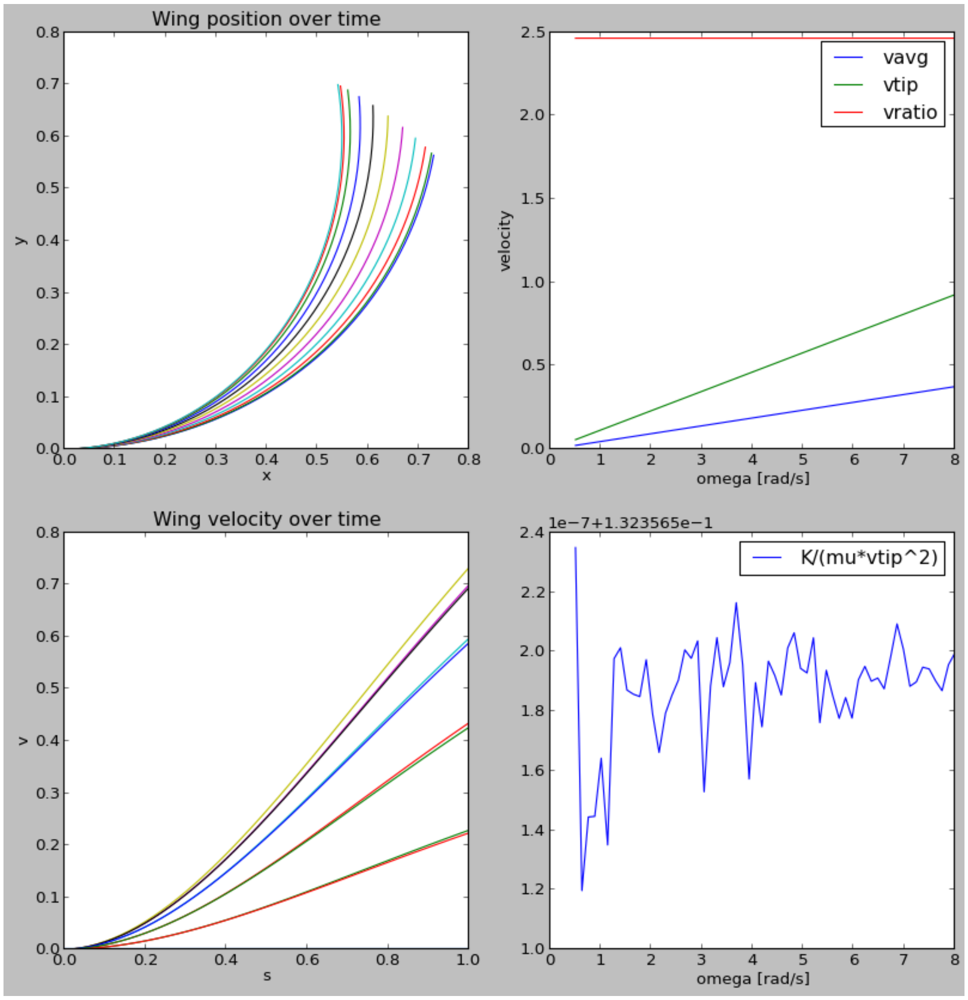


Appendix Figure I-3: Kinetic energy divided by $\mu_m v_{tip}^2$ is linear with length for fixed amplitude, static bending angle, and frequency

(same story)

I.2.2.1.5 K/mu-v² vs. omega

K/mu-v^2 is also independent of omega:



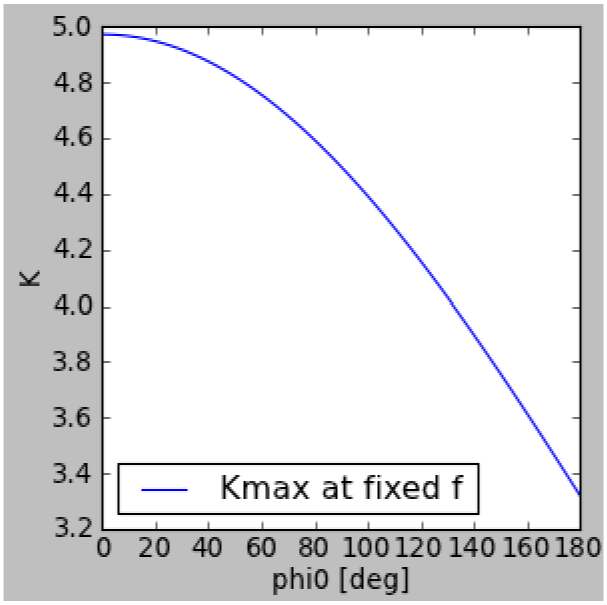
Appendix Figure I-4: Kinetic energy divided by $\mu_m v_{tip}^2$ is independent of frequency for fixed wing length, static bending angle, and amplitude

So,

$$\frac{K}{\mu v_{tip}^2} \propto L$$

Section I.2.2.2 K-dependence on ϕ_0 for fixed f: Deriving f_{resCor}

Using a similar approach to the last section, we could find that for fixed f as well as the other variables, K_{max} depends on ϕ_0 like $\left(1+2.0239\text{e-}2\phi_0^2+2.4241\text{e-}4\phi_0^4\right)^{-2}$:



Appendix Figure I-5: Max beam kinetic energy vs. bending angle shows a curious dependence (fixed f, L, A_r, and μ_m; all set to 1)

Instead, this section will take a more roundabout way to derive this same fact that settles on a useful name for this correction factor.

Since $U_{\max} = \tfrac{1}{2} EI \dfrac{A_r^2}{L} 3.09059 = \tfrac{1}{2} EI \dfrac{A_r^2}{L} \dfrac{\lambda^4}{4}$ is constant in ϕ_0 and f_0, and the max potential energy equals the max kinetic energy at resonance, therefore the max kinetic energy must also be constant in ϕ_0 and f_0 (for a fixed L & A_r).

However, the max kinetic energy turns out to be a function of both of these variables. We've already mentioned that $K_{\max} \propto f^2$, and by fixing f, L, and A_r we can rerun these calculations to determine that K is a decreasing function of ϕ_0. Therefore, for kinetic energy to be constant, the resonant frequency must be an increasing function of ϕ_0.

Encompass this dependence on frequency with a "ϕ_0-dependent resonant frequency correction factor", $f_{resCor}\left(\phi_0\right)$ defined so that

$$f = \frac{\lambda^2}{2\pi}\frac{1}{L^2}\sqrt{\frac{EI}{\mu_m}}f_{resCor}\left(\phi_0\right)$$

Constant max-kinetic energy at all ϕ_0 requires:

$K_{\max}\left(0, f_0\right) = K_{\max}\left(\phi_0, f_0 \times f_{resCor}\left(\phi_0\right)\right)$

and since $K \propto f^2$,

$K_{\max}\left(\phi_0, f_0 \times f_{resCor}\left(\phi_0\right)\right) = K_{\max}\left(\phi_0, f_0\right) \times \left(f_{resCor}\left(\phi_0\right)\right)^2$

so that $f_{resCor}\left(\phi_0\right) \equiv \sqrt{\frac{K_{\max}\left(0, f_0\right)}{K_{\max}\left(\phi_0, f_0\right)}}$ (as long as the peak potential energy is ϕ_0 independent, which is true if the bending shape and $\varepsilon_0 = 0$ is correct)

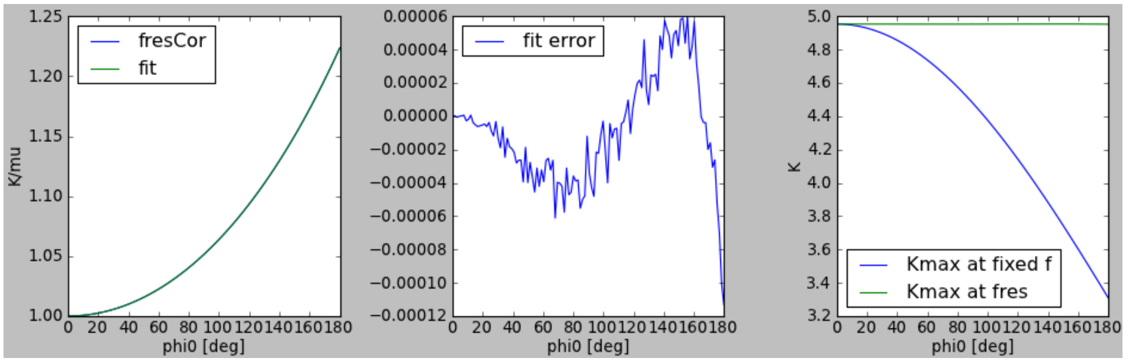
Result: from using function `derive_fresCor()` from `RSVelocityWork.py`, $f_{resCor}(\phi_0)$ turns out to be equal to:

fresCor = 1+(0.0202390368072*phi0^2+0.000242411506101*phi0^4)

(where phi0 is in radians).

The strictly even poly fit polynomial makes sense since the f_{res} correction should be symmetric about ϕ_0 = 0.

This function also plots the actually derived f_{resCor} along with this fit polynomial, the error between the two, and the verification that this resonant frequency does lead to constant kinetic energy, as required:



Appendix Figure I-6: Using the function `derive_fresCor()` from the program in Appendix J

Section I.2.2.3 Check: Energy balance for straight beam:

With no static bending angle (ϕ_0 = 0) we found in the last section that

$K_{\max}$=4.942×$\mu_m f^2 L^3 A_r^2$

Setting $U_{\max} = K_{\max}$ at resonance leads to the resonant frequency

$$\frac{3.0906}{2} EI \frac{A_r^2}{L} = 4.942\mu_m f^2 L^3 A_r^2$$

$$f = \sqrt{\frac{3.0906}{4.942\times 2}}\frac{1}{L^2}\sqrt{\frac{EI}{\mu_m}} = 0.5592\frac{1}{L^2}\sqrt{\frac{EI}{\mu_m}}$$

Check: Compare to $\frac{1.875^2}{2\pi} = 0.5595$. → Difference < 0.04%

Section I.2.2.4 RMS Velocity Compared to Kinetic Energy and a Straight Beam

For the straight beam, in Appendix D we found that the transverse displacement amplitude was given by

$\hat{w}(s) = A_1\left[\cosh\frac{\lambda}{L}s - \cos\frac{\lambda}{L}s + \frac{\cos\lambda + \cosh\lambda}{\sin\lambda + \sinh\lambda}\left(\sin\frac{\lambda}{L}s - \sinh\frac{\lambda}{L}s\right)\right]$. Referring to Section 2.2.6, A_1 is related to the RS mode's A_{r0} by $A_{r0} = 2\frac{A_1}{L}$, meaning that we could also write the amplitude as a function of arclength as

$\hat{w}(s)=\frac{A_{r0}L}{2}\left[\cosh\frac{\lambda}{L}s-\cos\frac{\lambda}{L}s+\frac{\cos\lambda+\cosh\lambda}{\sin\lambda+\sinh\lambda}\left(\sin\frac{\lambda}{L}s-\sinh\frac{\lambda}{L}s\right)\right]$. For harmonic motion with this amplitude, the maximum velocity in time is $v_{\max}(s)=\hat{w}(s)\cdot\omega$. This means the time-max spatial RMS velocity for the straight beam is:

$$v_{rms}\equiv\sqrt{\frac{\int_0^L\left(\omega\hat{w}(s)\right)^2 ds}{L}}=\omega\frac{A_{r0}L}{2}\sqrt{\frac{\int_0^L\left[\cosh\frac{\lambda}{L}s-\cos\frac{\lambda}{L}s+\frac{\cos\lambda+\cosh\lambda}{\sin\lambda+\sinh\lambda}\left(\sin\frac{\lambda}{L}s-\sinh\frac{\lambda}{L}s\right)\right]^2 ds}{L}}$$

This integral is nondimensionalized by the u-substitution $u\equiv\frac{s}{L}$, leading to:

$v_{rms}=\omega\frac{A_{r0}L}{2}\sqrt{\int_0^1\left[\cosh\lambda u-\cos\lambda u+\frac{\cos\lambda+\cosh\lambda}{\sin\lambda+\sinh\lambda}\left(\sin\lambda u-\sinh\lambda u\right)\right]^2 du}$, which evaluates to unity, meaning

$\therefore v_{rms}=\frac{A_{r0}\omega L}{2}$ (for a straight beam).

Numerically, the program in Appendix J finds that repeating the calculation for a curved beam modifies this same formula by f_{resCor}, to the time-max spatial RMS velocity:

$$v_{rms}=\frac{A_{r0}\omega L}{2f_{resCor}}$$

Comparing with the max kinetic energy, $K_{\max}=\frac{A_{r0}^2\omega^2L^3}{8f_{resCor}^2}\left(\rho_m Bh\right)$, and substituting $m_{wing}=\rho_m BhL$ confirms that the max kinetic energy is related to the time-max spatial RMS velocity by the familiar $K=\frac{1}{2}mv^2$.

Appendix J Python Program to Derive the RS mode Energy relations and Plots

```
# -*- coding: utf-8 -*-
"""
Created on Fri Apr 19 16:07:25 2013

@author: Minnick
"""
def rRS(phi0,L,Ar = 0, z = 0, ep0ByAr=0.00, numintervals=100):
    '''Returns the wing position array r(s) for a given Ar & z, where s
has numintervals+1 values from 0 to L

    phi0: full curvature angle in deg.  phi0 = c0*L
    L: wing length
    Ar: relative amplitude.  Ar = 2*A1/L --> A1 = Ar*L/2.
    z: offset from the neutral axis to compute
    ep0ByAr: longitudinal strain divided by Ar.
    numintervals: number of s-intervals to take from 0 to L.  (Points =
numintervals+1)
    '''
    import scipy as sp
    from scipy import cos
    from scipy import cosh
    from scipy import sin
    from scipy import sinh

    N = numintervals
    r=sp.empty((N+1,2))
    c0 = phi0*sp.pi/180/L
    Ds = L*1.0/N

    ep0= ep0ByAr*Ar

    #s = sp.arange(Ds/2, L+Ds/2, Ds)
    s = sp.linspace(Ds,L-Ds, N)
    r[0,:]=(0,z)

    #print s
    lam=1.8751
    Clam = (cos(lam)+cosh(lam))/(sin(lam)+sinh(lam))
    Lastbeta = 0
    for n,sval in enumerate(s):
        slam = sval*lam/L
        beta = c0*sval +Ar/2.*lam*(sinh(slam)+sin(slam)+Clam*(cos(slam)-
cosh(slam)))
        c = c0 + Ar/2.*lam**2/L*(cosh(slam)+cos(slam)+Clam*(-sin(slam)-
sinh(slam)))
        #beta = c0*sval +A*lam/L*(2*(slam+slam**4/120)+Clam*(-1*slam**2-
2*slam**6/720))
        #r[n+1,:]=r[n,:]+(1+ep0-z*c)*sp.array([cos(beta), sin(beta)])*Ds
#No chain rule for the z-part
```

```
        r[n+1,:]=r[n,:]+(1+ep0)*sp.array([cos(beta), sin(beta)])*Ds# +
sp.array([sin(beta)-sin(Lastbeta), -cos(beta)+cos(Lastbeta)])#Chain rule
for the z-part

    #z-correction part when using chain rule part
    for n,sval in enumerate(s):
        slam = sval*lam/L
        beta = c0*sval +Ar/2.*lam*(sinh(slam)+sin(slam)+Clam*(cos(slam)-
cosh(slam)))
        r[n+1,:]=r[n+1,:]-z*sp.array([sin(beta), 1.-cos(beta)])# +
sp.array([sin(beta)-sin(Lastbeta), -cos(beta)+cos(Lastbeta)])#Chain rule
for the z-part


    #r = sp.transpose(r)
    #RotMatrix = sp.array([[sp.cos(Rotation), -
sp.sin(Rotation)],[sp.cos(Rotation),-sp.sin(Rotation)]])
    #r = RotMatrix*r
    #r = sp.transpose(r)
    #print r
    return r

def rRS_Cyl(phi0,L,Ar = 0, z = 0, ep0ByAr=0.00, numintervals=100):
    import scipy as sp
    from scipy import cos
    from scipy import cosh
    from scipy import sin
    from scipy import sinh

    N = numintervals
    r=sp.empty((N+1,2))
    c0 = phi0*sp.pi/180/L
    Ds = L*1.0/N

    ep0= ep0ByAr*Ar

    #s = sp.arange(Ds/2, L+Ds/2, Ds)
    s = sp.linspace(Ds,L-Ds, N)
    dw=sp.empty(N+1)
    du=sp.empty(N+1)
    r0 = r*0
    rUW = r0*0
    #print s
    lam=1.8751
    Clam = (cos(lam)+cosh(lam))/(sin(lam)+sinh(lam))
    for n,sval in enumerate(s):
        slam = sval*lam/L
        dbeta = Ar/2.*lam*(sinh(slam)+sin(slam)+Clam*(cos(slam)-
cosh(slam)))
        beta = c0*sval +Ar/2.*lam*(sinh(slam)+sin(slam)+Clam*(cos(slam)-
cosh(slam)))
        beta0 = c0*sval
        r0[n+1,:] = r0[n,:]+sp.array([cos(beta0), sin(beta0)])*Ds
        #beta = c0*sval +A*lam/L*(2*(slam+slam**4/120)+Clam*(-1*slam**2-
2*slam**6/720))
        dw[n+1] = dw[n]+sin(dbeta)*Ds
        du[n+1] = du[n]-(1-cos(dbeta))*Ds
        rUW[n+1] = r0[n+1]+sp.array([cos(beta0)*du[n+1]-
sin(beta0)*dw[n+1], sin(beta0)*du[n+1]+cos(beta0)*dw[n+1]])

        r[n+1,:]=r[n,:]+(1+ep0-z*c0*sval)*sp.array([cos(beta),
sin(beta)])*Ds
    #beta0 = c0*sval

    return rUW, r


def rRS_at_t(phi0, L, Ar, omega, t, numintervals=50):
    from scipy import sin
    return rRS(phi0, L, Ar*sin(omega*t), numintervals=numintervals)

def vmag_at_t(phi0,L,Ar,omega,t = 0, numintervals=50, dt=1e-11):
    ''' returns the speed as a function of time t; if no speed is given,
returns max speed. '''
    from scipy import linspace
    from scipy import sqrt
    vVec = (rRS_at_t(phi0,L,Ar,omega,t+dt/2, numintervals=numintervals)-
rRS_at_t(phi0,L,Ar,omega,t-dt/2, numintervals=numintervals))/dt
    vMag = sqrt(vVec[:,0]**2+vVec[:,1]**2)
    N = len(vMag)
    s = linspace(0,L,N)
    return s, vMag

def arclength(r):
    ''' given a sequence of connected (x,y) as an array, calculates the
arclength of the object '''
    from scipy import sqrt
    x = r[:,0]
    y = r[:,1]
    S = 0
    for i in range(len(x)-1):
        Dx = x[i+1]-x[i]
        Dy = y[i+1]-y[i]
        S+=sqrt(Dx**2+Dy**2)
    return S


def derive_fits_vs_Ar(phi0):
    ''' outputs fits to displacement Dr(s/L,z=0,Ar) given phi0'''
    from scipy.optimize import curve_fit
    from scipy import linspace
    from scipy import empty
    from scipy import sin
#    from scipy import array
    L=1.
    def quartic(x,a,b,c,d):
        return(a*x**4+b*x**3+c*x**2+d*x**1)

    def quartic0(x,c1,c2,c3,c4):
        return(c1*x+c2*x**2+c3*x**3+c4*x**4)

    def quintic0(x,c1,c2,c3,c4,c5):
        ''' quintic function with y-intercept = 0'''
        return(c1*x+c2*x**2+c3*x**3+c4*x**4+c5*x**5)

    def quintic(x,c0,c1,c2,c3,c4,c5):
        ''' quintic function '''
        return(c0+c1*x+c2*x**2+c3*x**3+c4*x**4+c5*x**5)

#    def cubic(x,a,b,c,d):
#        return(a*x**3+b*x**2+c*x+d)

#    def quadratic(x,a,b,c):
#        return(a*x**2+b*x+c)
    def sinusoid(x,amp,omega,x0):
        return amp*sin(omega*(x-x0))

    def quartic2(x,c0,c1,c2,c3,c4):
        return(c0+c1*x+c2*x**2+c3*x**3+c4*x**4)

    def polyfit_string(FitDegree):
        if(FitDegree==4): return
'{}{:+}*Ar{:+}*Ar**2{:+}*Ar**3{:+}*Ar**4'
        if(FitDegree==5): return
'{}{:+}*Ar{:+}*Ar**2{:+}*Ar**3{:+}*Ar**4{:+}*Ar**5'

    sFitFunc = quintic0
    NP = 5 #Number of fit parameters in the fit to x & y

    print '-'*10+'fit starting'
    NArPts = 400  #Points to use for the fit to the fit parameters
    NSPts = 600  #Number of s-intervals to use when generating the r's
    r0 = rRS(phi0,L,0,numintervals=NSPts)
    s = linspace(0,L,len(r0[:,0]))
    xvar = 'Ar'
    SweepParam=linspace(-0.7,0.7, NArPts)
    fitvals = empty((NArPts,NP))
    fitvalsy = empty((NArPts,NP))
    for i,Ar in enumerate(SweepParam):
        rNow = rRS(phi0,L,Ar,numintervals=NSPts)-r0
        popt,pcov = curve_fit(sFitFunc, s, rNow[:,0])
        fitvals[i,:]=popt
        popt,pcov = curve_fit(sFitFunc, s, rNow[:,1])
        fitvalsy[i,:]=popt

    #print fitvals
    from matplotlib import pyplot as plt
    #from matplotlib import plot
    figrows = 4
    figcols = NP
    fig = plt.figure(figsize=(4*figcols,4*figrows))
    p=[]
    fitfit=[]
    fitfity=[]
    fitfunc=quintic
    fitfuncstr=polyfit_string(NP)

    ''' fit and plot x-fit parameters '''
    for i in range(NP):
        p.append(fig.add_subplot(figrows,figcols,i+1))
        p[i].set_xlabel(xvar)
        p[-1].set_ylabel('x fit parameter')
        p[i].plot(SweepParam, fitvals[:,i], label='cx{}'.format(i+1))

        popt,pcov = curve_fit(fitfunc, SweepParam, fitvals[:,i])
        fitfit.append(popt)
        #amp,omega,x0=popt
        p[i].plot(SweepParam, fitfunc(SweepParam, *popt), label='fit')
        p[i].legend(loc=0)
        print 'cx{} = '.format(i+1)+fitfuncstr.format(*fitfit[i])

    ''' show x-fit parameter errors '''
    for i in range(NP):
        p.append(fig.add_subplot(figrows,figcols,len(p)+1))
        p[-1].set_xlabel(xvar)
        p[-1].set_ylabel('x fitfit error')

        params=fitfit[i]
        p[-1].plot(SweepParam, fitfunc(SweepParam, *params)-
fitvals[:,i], label='cx{}Er'.format(i+1))
        p[-1].legend(loc=0)


    ''' fit and plot y-fit parameters '''
    for i in range(NP):
        p.append(fig.add_subplot(figrows,figcols,len(p)+1))
        p[-1].set_xlabel(xvar)
        p[-1].set_ylabel('y fit parameter')
        p[-1].plot(SweepParam, fitvalsy[:,i-NP],
label='cy{}'.format(i+1))

        popt,pcov = curve_fit(fitfunc, SweepParam, fitvalsy[:,i])
        fitfity.append(popt)
        #a,b,c,d=popt
        p[-1].plot(SweepParam, fitfunc(SweepParam, *popt), label='fit')
        print 'cy{} = '.format(i+1)+fitfuncstr.format(*fitfity[i])
        p[-1].legend(loc=0)

    for i in range(NP):
        p.append(fig.add_subplot(figrows,figcols,len(p)+1))
        p[-1].set_xlabel(xvar)
        p[-1].set_ylabel('y fitfit error')

        params=fitfity[i]
        p[-1].plot(SweepParam, fitfunc(SweepParam, *params)-
fitvalsy[:,i], label='cy{}Er'.format(i+1))
        p[-1].legend(loc=0)

#    for i in range(len(p),len(p)+NP):
#        p.

    opXstr='Dx ='
    opYstr='Dy ='
    for i in range(1,NP+1):
        operator = '+' if i>1 else ''
        opXstr+=operator+'cx{}*S**{}'.format(i,i)
        opYstr+=operator+'cy{}*S**{}'.format(i,i)
    print opXstr
    print opYstr
#    print 'Dx = cx4*s**4+cx3*s**3+cx2*s**2+cx1*s'
#    print 'Dy = cy4*s**4+cy3*s**3+cy2*s**2+cy1*s'
    plt.tight_layout()
    plt.show()

def compare_fit_to_iteration(phi0=90, L=1):
    import matplotlib.pyplot as plt

    #from scipy import array
    from scipy import linspace
    #from scipy import pi
    NSPts = 300

    r0 = rRS(phi0, L, numintervals=NSPts)

    s = linspace(0,L,len(r0[:,0]))

    fig = plt.figure(figsize=(24,12))
    PlotGridWidth=2
    PlotGridHeight=1
    p=[]

    '''Fit comparison'''
    p.append(fig.add_subplot(PlotGridHeight,PlotGridWidth,len(p)))
    p[-1].set_xlabel('x')
    p[-1].set_ylabel('y')
    p[-1].set_title('Positions of deformed wing; full = blue, fit =
green')
    p[-1].set_xlim((0,L))
    p[-1].set_ylim((-L,0))
    p.append(fig.add_subplot(PlotGridHeight,PlotGridWidth,len(p)))
    p[-1].set_xlabel('s')
    p[-1].set_ylabel('FitErr')
    p[-1].set_title('Error of fit; blue = x, green = y')
    for Ar in linspace(-.5, .5, 10):
        r = rRS(phi0, L, Ar, numintervals=NSPts)
        #Dr = r-r0

        #x,y = Dr_fit(Ar, s)
        DrFit = Dr_fit(s, Ar)
        rFit = DrFit + r0

        p[-2].plot(r[:,0], r[:,1], 'b')
        p[-2].plot(rFit[:,0], rFit[:,1], 'g')

        Err = rFit-r
        p[-1].plot(s, Err[:,0], 'b')
        p[-1].plot(s, Err[:,1], 'g')

    fig.tight_layout()
    plt.show()


def r_cc_point(c0,s,z=0):
    ''' returns the point that (s,z) maps to on a constant curvature arc
with curvature c0 '''
    from scipy import sin
    from scipy import cos
    #from scipy import array
    from scipy import empty
    x = (1/c0-z)*sin(s*c0)
    y = z+(1/c0-z)*(1-cos(s*c0))
    r = empty((len(s),2))
    r[:,0],r[:,1]=x,y
    return r


def r_cc(phi0,L,z=0, numpoints=101):
    ''' return a set of points on a constant curvature arc '''
    from scipy import pi
    from scipy import linspace
    phi0rad = phi0*pi/180.
    c0 = phi0rad/L
    s = linspace(0,L,numpoints)
    return r_cc_point(c0,s,z)

def r_fit(Ar, s, z, L=1, phi0=90):
    ''' returns the x,y points corresponding to Ar, s, & z for the
current fit.

    s may be an array, but z should be a scalar

    The fit itself is for fixed L & phi0
    '''
    from scipy import empty
    from scipy import pi
    from scipy import sin
    from scipy import cos
    phi0rad = phi0*pi/180.
    c0 = phi0rad/L

    r0 = r_cc_point(c0,s,0) #position of undeformed arc
    Dr0 = Dr_fit(s, Ar) #deformation of centre-line
    betaNow=beta(s/L,Ar,phi0)
    #beta0 = beta(s/L,0,phi0)

    rzExtra = empty([len(s),2])
    rzExtra[:,0]=z*(-sin(betaNow))
    rzExtra[:,1]=z*(cos(betaNow))

    return r0+Dr0+rzExtra

def r_disp_fitt(Ar, s, z, L=1, phi0=90):
    ''' returns the x,y displacements from Ar = 0 corresponding to Ar,
s, & z for the current fit.

    s may be an array, but z should be a scalar

    The fit itself is for fixed L & phi0
    '''
    from scipy import empty
    from scipy import pi
    from scipy import sin
    from scipy import cos
    phi0rad = phi0*pi/180.
    c0 = phi0rad/L

    r0 = r_cc_point(c0,s,0) #position of undeformed arc
    Dr0 = Dr_fit(s, Ar) #deformation of centre-line
    betaNow=beta(s/L,Ar,phi0)
    #beta0 = beta(s/L,0,phi0)

    rzExtra = empty([len(s),2])
    rzExtra[:,0]=z*(-sin(betaNow))
    rzExtra[:,1]=z*(cos(betaNow))

    return Dr0+rzExtra

def beta(sByL, Ar=0, phi0=90):
    from scipy import pi
    from scipy import cos
    from scipy import cosh
    from scipy import sin
    from scipy import sinh

    phi0rad = phi0*pi/180.
    lam=1.8751
    Clam = (cos(lam)+cosh(lam))/(sin(lam)+sinh(lam))
    slam = sByL*lam
    beta = sByL*phi0rad
+Ar/2.*lam*(sinh(slam)+sin(slam)+Clam*(cos(slam)-cosh(slam)))
```

```
    return beta

def Dbeta(sByL, Ar, phi0=90):
    return beta(sByL, Ar, phi0)-beta(sByL,0,phi0)

def calc_symbolic_delta_U():
    from sympy import symbols
    from sympy import diff
    from sympy import cos
    from sympy import cosh
    from sympy import sin
    from sympy import sinh
    from sympy import integrate
    from sympy import Eq
    from sympy.solvers.solvers import nsolve
    from sympy.solvers.solvers import solve
    from sympy import simplify

    L, Ar, s, lam = symbols('L Ar s lam', positive=True)
    sByL = s/L
    Clam = (cos(lam)+cosh(lam))/(sin(lam)+sinh(lam))
    slam = sByL*lam
    Dbeta = Ar/2.*lam*(sinh(slam)+sin(slam)+Clam*(cos(slam)-cosh(slam)))
    Dc = diff(Dbeta, s)
    DcDiff = diff(Dc, s)


    #1----------Solve for lam
    expr  = DcDiff.subs({s:L, Ar:1})  #hard mode
    expr = simplify(expr*L**2)  #need to remove the L to enable solving

    print 'Expression to solve for lambda: expr:',  expr

    lamSoln = nsolve(expr, lam, 1.9, tol=1e-9)
    print 'Found lamSoln =', lamSoln
    print 'using lam = 1.8751:', expr.subs({lam:1.8751})
    print 'using lam = lamSoln:', expr.subs({lam:lamSoln})

    #2----------Solve for U
#    LVal=1
#    ArVal=1
#    DcSub = Dc.subs({lam:lamSoln, L:LVal, Ar:ArVal})
#    print DcSub
#    Dc2Sub = simplify(DcSub**2)

    lam = lamSoln
    Clam = (cos(lam)+cosh(lam))/(sin(lam)+sinh(lam))
    f= lambda S: 1/2.*lam**2*(cosh(lam*S)+cos(lam*S)+Clam*(-sin(lam*S)-
sinh(lam*S)))
    print f(1)
#    #U = integrate(DcSub**2, (s,0,LVal))  #Doesn't work
    from scipy import linspace
    from scipy import vectorize
    fv = vectorize(f)
    N = 30
    SEndPoints = linspace(0,1, N+1)
    S = (SEndPoints[0:-1]+SEndPoints[1:])/2.  #midpoints
    Ds=1/float(N)
    dF = fv(S)

#    dF=sMPs*0 #initialize
#    for (i,sval) in enumerate(sMPs):
#        dF[i] = Dc2Sub.subs({s:sval})*Ds
    f2Int = Ds*sum(dF**2)
    print 'Constant for the DeltaU expression =', f2Int


    #print 'U =', U

    #print Dbeta
    #print Dc**2
    #IntDC2 = integrate(Dc**2, (s, 0, L))
    #print 'Integral of (deltaC/(Ar/2))^2 from s = 0 to L =', IntDC2

    return


def Dr_fit(s,Ar,L=1):
    ''' returns x,y displacements of centre-line for a given Ar & s

    This fit is for:
    phi0=90
    '''
    from scipy import empty
    S= s/float(L) #dimensionless arclength
    Dr=empty((len(s),2))


    ''' phi0 = -90, epsilon0 = 0 '''
    cx1 = -1.45736312265e-06+0.00661860003827*Ar+0.00457111307058*Ar**2-
0.0148675258367*Ar**3+0.00877461543996*Ar**4+0.000591190828005*Ar**5
    cx2 = 1.57713940247e-05-0.0875967393542*Ar-
0.0356116187383*Ar**2+0.169499584388*Ar**3-0.11535586392*Ar**4-
0.00245614556228*Ar**5
    cx3 = -4.62539757505e-05+2.26925244918*Ar-2.12023952113*Ar**2-
0.517881899062*Ar**3+0.457728554782*Ar**4-0.0182093310178*Ar**5
    cx4 = 2.92897672562e-05-1.87806615128*Ar+2.97532556221*Ar**2-
0.0109868415584*Ar**3-0.451348618295*Ar**4+0.0879009244088*Ar**5
    cx5 = -1.14464059178e-06+0.448345774142*Ar-
1.10061447187*Ar**2+0.174197104622*Ar**3+0.12956024793*Ar**4-
0.051496085948*Ar**5
    cy1 = -3.08139675884e-08-0.00365210950858*Ar+0.0269630858174*Ar**2-
0.0169273678953*Ar**3-0.00407318078444*Ar**4+0.00263898095313*Ar**5
    cy2 = 3.57396161896e-06+1.83468607452*Ar-
0.36731980764*Ar**2+0.26240091504*Ar**3+0.0376800751839*Ar**4-
0.0311705859764*Ar**5
```

```
    cy3 = -3.033325547e-05-1.13190196682*Ar+1.68944872664*Ar**2-
1.4076808631*Ar**3-0.0606389966332*Ar**4+0.105043915957*Ar**5
    cy4 = 7.58208881938e-05-0.575655943153*Ar-
0.937567950184*Ar**2+1.57704797061*Ar**3-0.15722628534*Ar**4-
0.0882171676537*Ar**5
    cy5 = -3.97054148684e-05+0.410187725349*Ar+0.059811543127*Ar**2-
0.5183633958*Ar**3+0.122053142231*Ar**4+0.0188127891307*Ar**5
    Dx =cx1*S**1+cx2*S**2+cx3*S**3+cx4*S**4+cx5*S**5
    Dy =cy1*S**1+cy2*S**2+cy3*S**3+cy4*S**4+cy5*S**5



    Dr[:,0]=Dx+0
    Dr[:,1]=Dy+0
    return Dr*L

def Dr_offaxis_fit(s,z,Ar=0,L=1.,phi0=90.):
    ''' Uses the fit function Dr_fit to compute the displacement of an
on or off-axis point '''
    #from scipy import pi
    from scipy import cos
    from scipy import sin
    #phi0rad = phi0*pi/180.
    beta0 = beta(s/L, 0, phi0)
    beta1 = beta(s/L, Ar, phi0)
    Dr = Dr_fit(s, Ar, L)
    x = -z*(sin(beta1)-sin(beta0))
    y = z*(cos(beta1)-cos(beta0))
    Dr[:,0]+=x
    Dr[:,1]+=y
    return Dr

def KmaxByMu(phi0, L, Ar, omega, numintervals=50):
    ''' returns max kinetic energy over mu '''
    s, vmax = vmag_at_t(phi0,L,Ar,omega, numintervals=numintervals)
    Ds =s[2]-s[1]
    KmaxByMu = 0.5*sum(vmax**2)*Ds
    return KmaxByMu

def K_at_t(phi0, L, Ar, omega, t, numintervals=100):
    ''' returns K/mu at time t '''
    s, v = vmag_at_t(phi0,L,Ar,omega,t=t, numintervals=numintervals)
    Ds =s[2]-s[1]
    KByMu = 0.5*sum(v**2)*Ds
    return KByMu

def U_at_t(phi0, L, Ar, omega, t):
    ''' returns the potential energy(/EI) at time t '''
    from scipy import sin
    return 3.0906/2*(Ar*sin(omega*t))**2/L


def getz(r0cc,s,c0):
    from scipy import cos
    from scipy import sin
    beta0 = s*c0
    z0= -sin(beta0)*r0cc[:,0]+r0cc[:,1]*cos(beta0)+(1-cos(beta0))/c0
    return z0

def derive_fresCor():
    ''' Derives the f_resCor so that Um is constant in Ar for RS bending


    Result on 22 Apr using
    numintervals=800
    numphipoints=100
    --> fresCor = 1-(5.78534024314e-06*phi0**2)
    '''
    from scipy.optimize import curve_fit
    from scipy import sqrt
    from matplotlib import pyplot as plt
    #from matplotlib import plot
    fig = plt.figure(figsize=(12,4))
    p=fig.add_subplot(131)
    errplot=fig.add_subplot(132)
    Kplots = fig.add_subplot(133)


    from scipy import linspace
    from scipy import pi
    L = 1
    Ar = 1
    omega = 2*pi#*0.571

    numintervals=800
    numphipoints=120



    K0 = KmaxByMu(0,L,Ar,omega, numintervals=numintervals)

    def quadratic(x,c1,c2):
        return c1*x+c2*x**2
    #fitstring = '{}{:+}*phi0{:+}*phi0**2{:+}*phi0**3{:+}*phi0**4'
    def square(x,c):
        return c*x**2

    def poly25(x,c2,c5):
        return c2*x**2+c5*x**5
    def quintic(x,c0,c1,c2,c3,c4,c5):
        ''' quintic function '''
        return(c0+c1*x+c2*x**2+c3*x**3+c4*x**4+c5*x**5)

    def poly24(x,c2,c4):
        return c2*x**2+c4*x**4
    fitfunc = poly24
    #fitstring = '{}*phi0{:+}*phi0**2'  #quadratic
    #fitstring = '{}*phi0**2' #square
    #fitstring = '{}*phi0**2+{}*phi0**5' #poly25
    #fitstring =
'{}{:+}*phi0{:+}*phi0**2{:+}*phi0**3{:+}*phi0**4{:+}*phi0**5'
#fullQuintic
    fitstring = '{}*phi0**2{:+}*phi0**4' #poly24
#    def quadratic

    #return

    Sweeprange = linspace(0,180, numphipoints)
    fresCor=Sweeprange*0 #initialize
    Kmax=Sweeprange*0
    for (i,phi0) in enumerate(Sweeprange):
        #Umax = 3.0906/2*Ar
        Kmax[i] = KmaxByMu(phi0, L, Ar, omega,
numintervals=numintervals)
        fresCor[i]=sqrt(K0/Kmax[i])

    phi0List = Sweeprange*pi/180.
    popt, pcov = curve_fit(fitfunc, phi0List, fresCor-1)
    print 'fresCor = 1+('+fitstring.format(*popt)+')'
    fresfit = 1+fitfunc(phi0List, *popt)

#    K_with_fChange=Sweeprange*0
#    for (i,phi0) in enumerate(Sweeprange):
#        K_with_fChange[i]=KmaxByMu(phi0, L, Ar, omega*fresfit[i],
numintervals=numintervals)  #Detailed
    #optimized:
    K_with_fChange = Kmax*(fresfit**2)

    p.plot(Sweeprange,fresCor, label='fresCor')
    p.plot(Sweeprange,fresfit, label='fit')
    err=(fresfit-fresCor)
    errplot.plot(Sweeprange, err, label='fit error')
    errplot.set_xlabel('phi0 [deg]')
    errplot.legend(loc=0)
    p.set_xlabel('phi0 [deg]')
    p.set_ylabel('K/mu')
    p.legend(loc=0)

    Kplots.plot(Sweeprange, Kmax, label='Kmax at fixed f')
    Kplots.plot(Sweeprange, K_with_fChange, 'g', label='Kmax at fres')

    Kplots.set_xlabel('phi0 [deg]')
    Kplots.set_ylabel('K')
    Kplots.legend(loc=0)

    fig.tight_layout()
    plt.show()
```

```
def Dr_cyl_by_rho0(sByL, Ar, phi0, numpoints=50):
    from scipy import linspace
    from scipy import sin
    S = linspace(0, sByL, numpoints)
    from scipy import pi
    rho0 = 1/(phi0*pi/180)
    return -sum(sin(Dbeta(S, Ar, abs(phi0))))*sByL/float(numpoints)/rho0

def cylindrical_approx_velocity(phi0,L,Ar,omega,t = 0, numintervals=500,
dt=1e-11):
    ''' returns the speed as a function of time t; if no speed is given,
returns max speed. '''
    from scipy import linspace
    from scipy import sqrt
    from scipy import pi
    from scipy import sin
    c0 = phi0*pi/180./L
    sEndpoints = linspace(0,L,numintervals+1)
    #s = (sEndpoints[:-1]+sEndpoints[1:])/2.
    s = sEndpoints
    Ds = s[2]-s[1]
    betaDot = (beta(s/L, Ar*sin(omega*(t+dt)), phi0)-beta(s/L,
Ar*sin(omega*t), phi0))/dt
    vr=s*0
    vq=s*0
    for i in range(len(s)):
        vr[i] = sum(betaDot[:i])*Ds
        #vq[i] = sum(vr[:i])*Ds

    vq = vq*c0**1

    vMag = sqrt(vr**2+vq**2)
    #N = len(vMag)
    #s = linspace(0,L,N)
    return s, vMag

def fresCor(phi0deg):
    ''' expects phi0 in deg '''
    from scipy import pi
    phi0rad = phi0deg*pi/180
    return 1+(0.0202390368072*phi0rad**2+0.000242411506101*phi0rad**4)

def KmaxbyMufit(phi0deg, L, Ar, omega):
    ''' uses the fit function to return KmaxbyMu for given flapping
parameters '''
    return L/8.*(Ar*omega*L/fresCor(phi0deg))**2

def main_loop():
    #calc_symbolic_delta_U()



    phi0=-50.
```

```
#    derive_fits_vs_Ar(phi0)


    #compare_fit_to_iteration(phi0)
   # derive_fresCor()

    import matplotlib.pyplot as plt

    #from scipy import array
    from scipy import linspace
    from scipy import pi

#    derive_fresCor()
#    return
    #phi0=0

    L = 5
    Ar = .2
    NSPts =30
    #ZPts = 5
    c0 = (phi0*pi/180.)/L
    h = .04
    omega = 7*pi#*0.571

    #print K_at_t(phi0, L, Ar, omega, 0,
numintervals=500)/(Ar*omega*L/fresCor(phi0))**2
    N = 5000
    print KmaxByMu(phi0,L,Ar,omega,numintervals=N)
    print KmaxbyMufit(phi0,L,Ar,omega)
    #Extreme Displacements
    #phi0, ArLimit = 75, .7
    phi0, ArLimit = 90, .6
    #phi0, ArLimit = 105, .5
    #phi0, ArLimit = 120, .4
    #phi0, ArLimit = 140, .25
    #phi0, ArLimit = 0,  .5
    #phi0, ArLimit = 105, .6
  #  phi0, ArLimit = 75, .0
    #phi0, ArLimit = 90,0.0


 #   print (beta(1,ArSmall, phi0)-beta(1,0, phi0))**2/2
#    print Dr_cyl_by_rho0(1, ArSmall, phi0)

    rUW,r = rRS_Cyl(phi0, L, Ar)

    s = linspace(0,L,len(r[:,0]))
    #plt.plot(r[:,0], r[:,1], 'b', label = 'original formulation')
    #plt.plot(rUW[:,0], r[:,1], 'g', label = 'formulation using uw')
    #plt.legend(loc=0)
#    Dr1 = r1-r0

    #r0cc=[]
```

```
rA=[]

#plt.show()
Figsize = 5

fig1 = plt.figure(figsize = (Figsize,Figsize))
rRSPlot = fig1.add_subplot(1,1,1)
BoxSize = 1.05
Bot = -.2
rRSPlot.set_xlim((0,BoxSize))
rRSPlot.set_ylim((Bot,Bot+BoxSize))

h = .02


import scipy as sp
for Ar in linspace(-ArLimit,ArLimit,5):
    rTop =rRS(phi0, 1,Ar, z=h/2)
    rBot =rRS(phi0, 1,Ar, z=-h/2)
    xpts = sp.concatenate((rTop[:,0], rBot[::-1,0]))
    ypts = sp.concatenate((rTop[:,1], rBot[::-1,1]))
    #rRSPlot.plot(rTop[:,0], rTop[:,1])
    #rRSPlot.plot(rBot[:,0], rBot[:,1])
    extrax = []
    extray = []
    ArrayPts = 101

    ZigZags = False
    Bars = True

    if ZigZags:
        Zags = 100
        StepSize = ArrayPts/Zags
        for i in range(Zags/2):
            Index1 = 2*i*StepSize
            Index2 = (2*i+1)*StepSize
            extrax.append(rTop[Index1,0])
            extrax.append(rBot[Index2,0])
            extray.append(rTop[Index1,1])
            extray.append(rBot[Index2,1])
    elif Bars:
        TopNext=True
        for i in range(0,ArrayPts, 2):
            if(TopNext):
                TopNext=False
                extrax.append(rTop[i,0])
                extray.append(rTop[i,1])
                extrax.append(rBot[i,0])
                extray.append(rBot[i,1])
            else:
                TopNext=True
                extrax.append(rBot[i,0])
                extray.append(rBot[i,1])
```

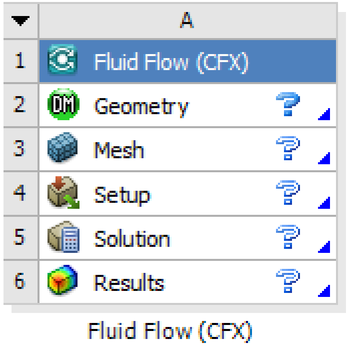
```
                    extrax.append(rTop[i,0])
                    extray.append(rTop[i,1])
        extrax.append(rTop[-1,0])
        extray.append(rTop[-1,1])
        xpts = sp.concatenate((xpts, sp.array(extrax)))
        ypts = sp.concatenate((ypts, sp.array(extray)))
        rRSPlot.plot(xpts, ypts)
        #rRSPlot.plot(extrax, extray)

    print 'Top of tip position = ', rTop[-1,:]
    print rTop[-1,0]


     #       plt.plot(s,z0-z)
            #rA.append(r_fit(Ar,s,z,L,phi0))
#            rA.append(Dr_offaxis_fit(s,z,Ar,L,phi0)+r0cc)
#            rRSPlot.plot(rA[-1][:,0], rA[-1][:,1])

#    for i in range(5):
#        z= 0.005*i
    fig1.patch.set_facecolor('white')
    plt.title('$\phi_0={}^o, {} = {}$'.format(phi0, 'A_{r0}', ArLimit))
    plt.savefig('RSPlot phi{} Ar{}.png'.format(phi0, ArLimit),
bbox_inches='tight', pad_inches=0, transparent=True, dpi=300)

    plt.show()

if __name__ == '__main__':
    main_loop()
```

Appendix K Tutorial for Creating the phi0g120 sims in ANSYS, using ANSYS 13

"phi0g120" in the simulation filename means that the static angle ϕ_0 for the geometry setup is 120^{o}.

Launch ANSYS Workbench.

Create a Fluid Flow (CFX) Analysis system.



▼	A		
1	Fluid Flow (CFX)		
2	Geometry	?	◢
3	Mesh	?	◢
4	Setup	?	◢
5	Solution	?	◢
6	Results	?	◢

Fluid Flow (CFX)

Edit geometry with design modeller.

Set units to mm.

Section K.1 Geometry part:

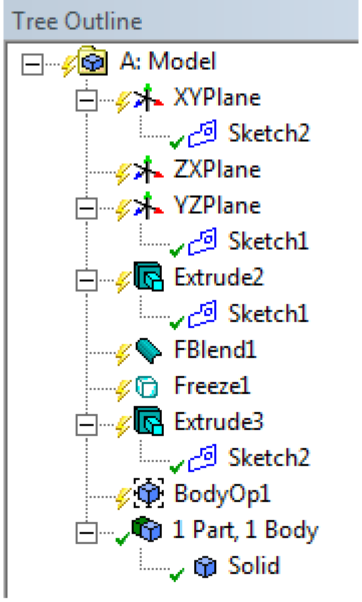
Overview:

We're going to make a wing with these dimensions:

1. L = 2 mm, $\phi_{0g} = 120°$, and $L = \rho_0 \phi_{0g}$
2. → $\rho_0 = \dfrac{L}{\phi_{0g}} = 0.95493\text{mm}$
3. $B/2$ = 0.5 mm
4. $h/2$ = 0.025 mm
5. “Blend” on edges: radius = 8 µm

And the wing will be placed in a cylindrical fluid domain with dimensions *Radius* = 20 mm, *Height* = 40 (*Height*/2 = 20 mm). This means the fluid domain extends for at least 10 wing lengths in all directions, except for the symmetry planes immediately adjacent to the wing.

At the end, the components of the geometry will look like this:

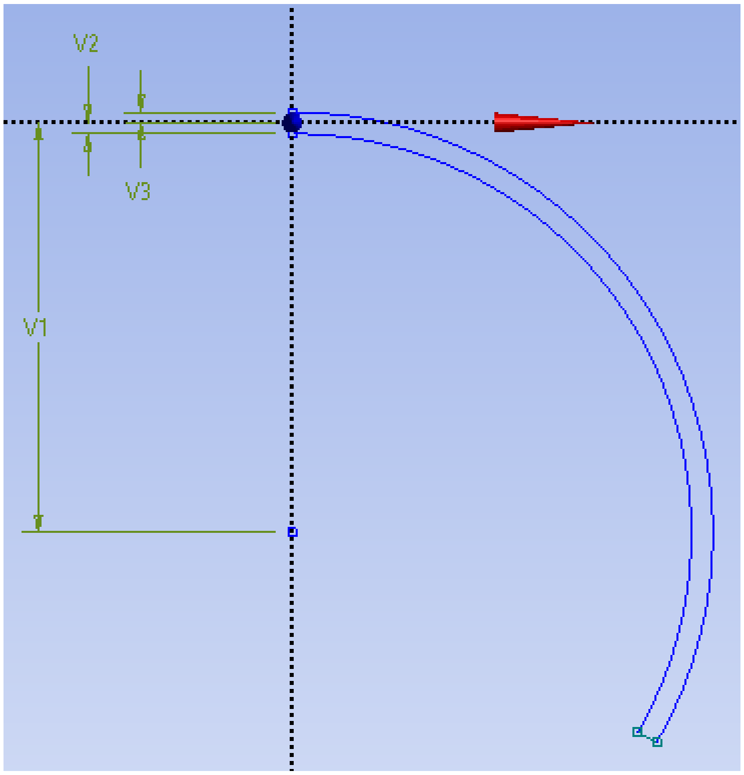


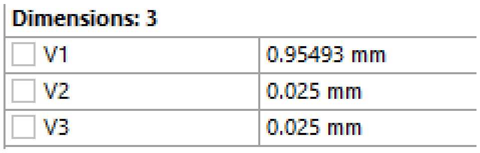
Tree Outline

- A: Model
  - XYPlane
    - Sketch2
  - ZXPlane
  - YZPlane
    - Sketch1
  - Extrude2
    - Sketch1
  - FBlend1
  - Freeze1
  - Extrude3
    - Sketch2
  - BodyOp1
  - 1 Part, 1 Body
    - Solid

Steps:

1. Make a sketch on the yz plane for the wing: Sketch1
   a. Draw the wing outline
      i. Finished Version:

ii. 

iii. 

Dimensions: 3	
□ V1	0.95493 mm
□ V2	0.025 mm
□ V3	0.025 mm

iv. Sizes are from this:

1. L = 2 mm, $\phi_{0g} = 120°$, and $L = \rho_0 \phi_{0g}$
2. $\rightarrow \rho_0 = \dfrac{L}{\phi_{0g}} = 0.95493\text{mm}$
3. $h/2$ = 0.025 mm

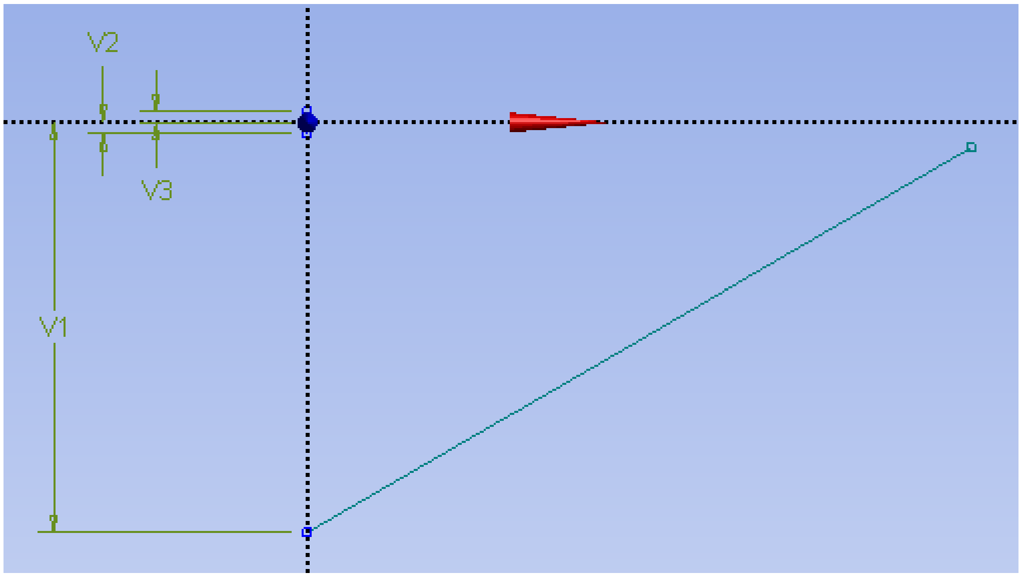
b. To make the sketch:

i. Draw a construction point on the negative z-axis a distance ρ_0 from the origin.

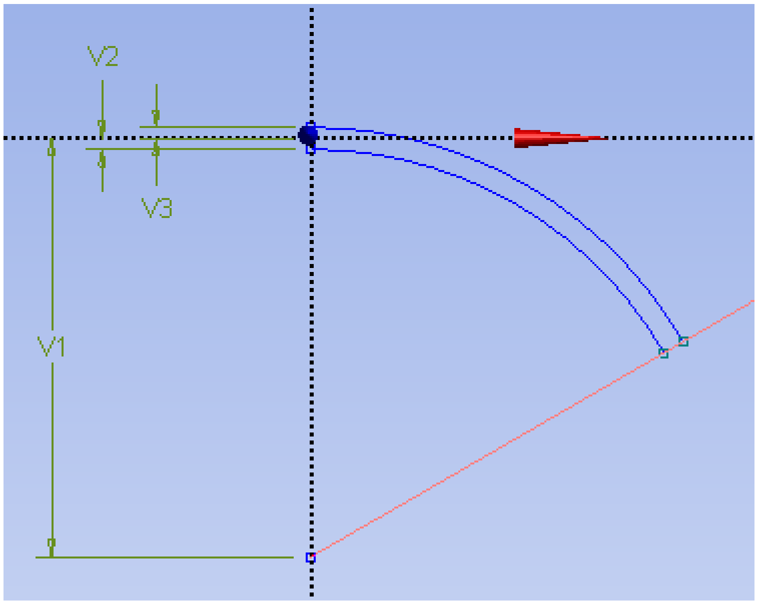
ii. Draw lines from the origin down and up by $h/2$

1. Do this by placing the point and drawing the lines to any distance, then specifying their dimensions.

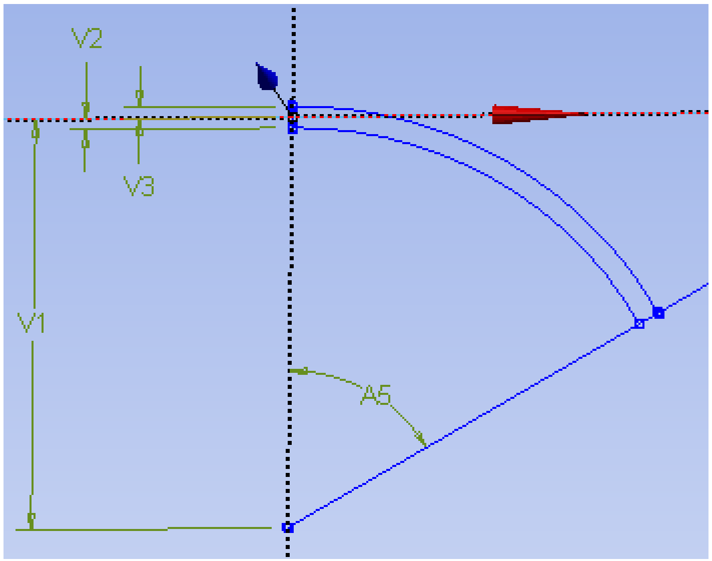
iii. Draw a line at a non-horizontal, non-vertical angle to the right from the construction point to serve as a guide

iv. 

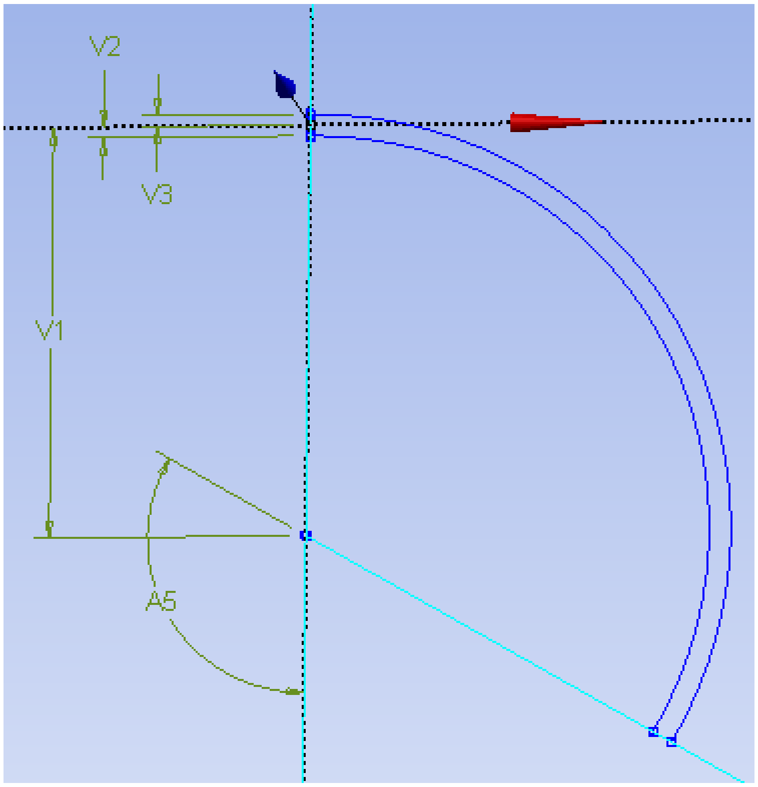
v. Draw circular arcs "Arc by Center" with the construction point as centre and the ends of the vertical lines from the origin as start points that stop on the guide line:

vi. 

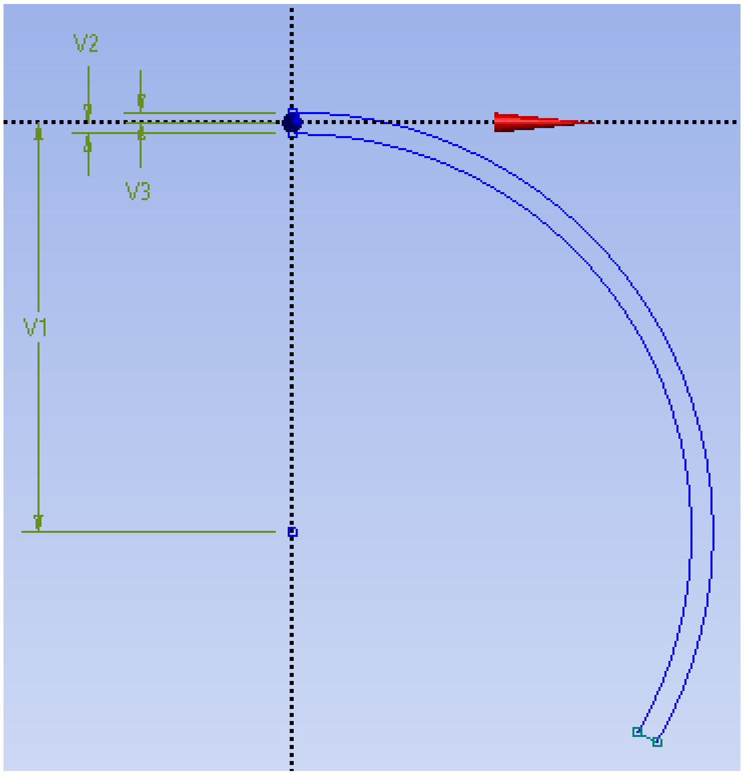
vii. Use an Angle dimension to specify that the guide line should be at 120° from the vertical

viii. 

1. → After setting A5 to 120°:

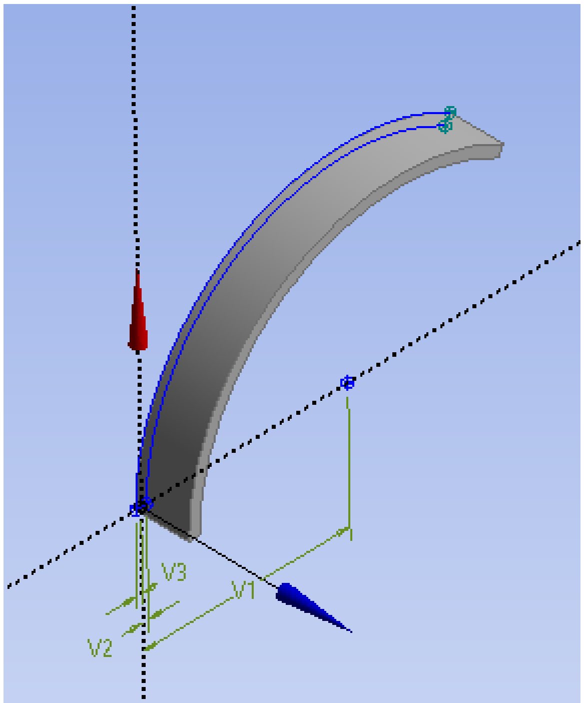
2. 

ix. Delete the guide line from the construction point, and make a final line connecting the open end of the arc.

x. 

2. Extrude the wing shape from sketch 1

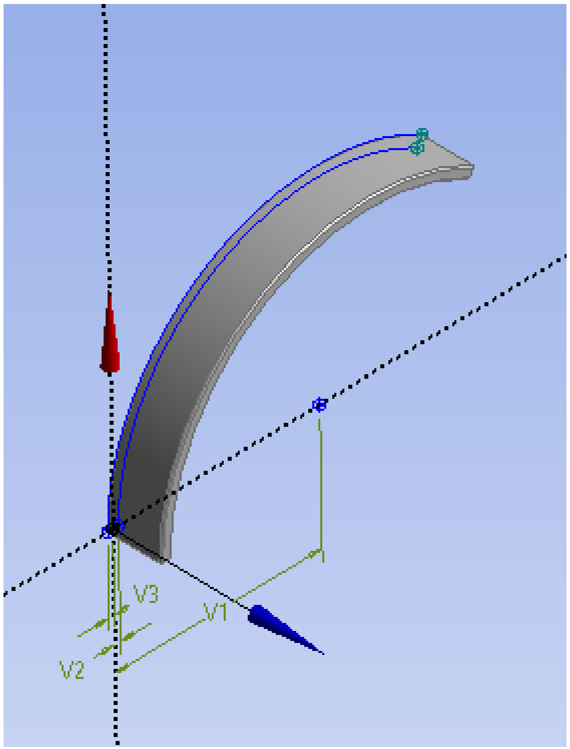
   c. A distance *B*/2 = 0.25 mm (in the +x direction; "normal")

d. 

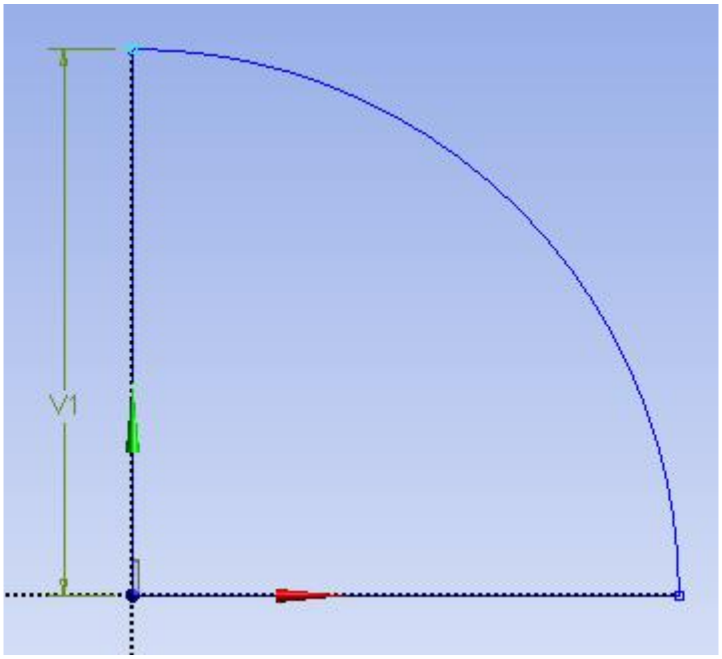
3. Blend the external edges of the wing to smooth it for easier meshing

    e. Use all five external edges (not the ones on the yz or xz plane because those are going to be mirror planes)

    f. Fixed radius = *L*/250 = 0.008 mm

g. 

4. Freeze the wing
   a. Select it in the tree outline under 1 Part, 1 Body
   b. Tools Menu: Freeze
5. Make a sketch on the xy plane for the edge of the cylindrical fluid domain, which is a quarter-circle

h. 

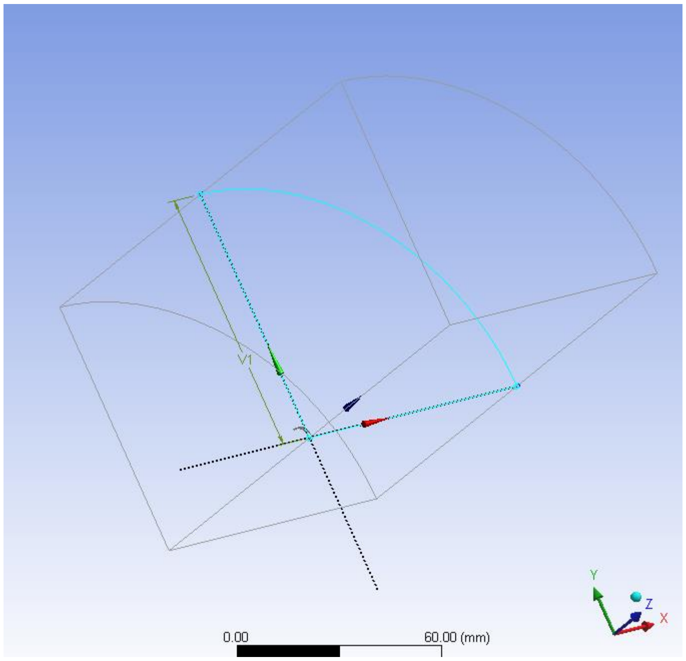
i. Line up from origin to edge a distance of *Radius* = 10*L* = 20 mm

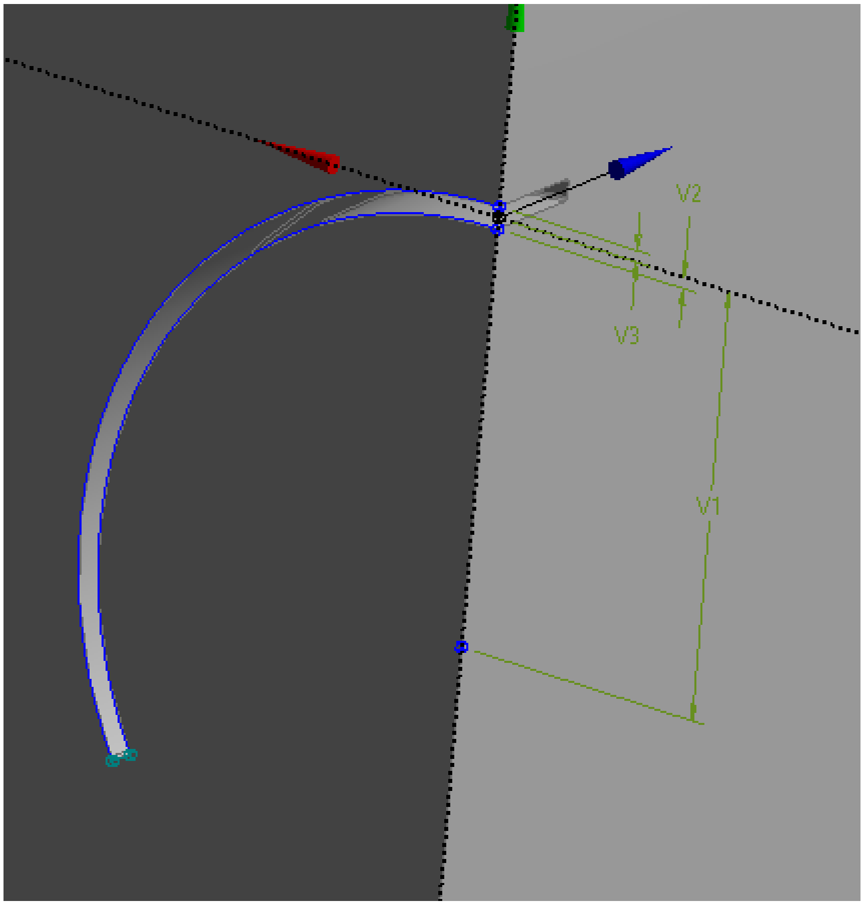
j. Circular arc down to the right by 90^{o} angle

k. Line closing the edge

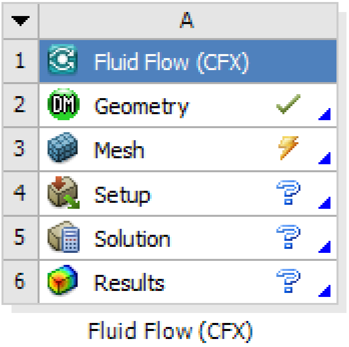
6. Extrude the fluid domain

l. both directions, by *Height*/2 = 20 mm in each direction

m. 

7. body op: cut the wing from the fluid domain
    a. Create: Body Operation
    b. Type: Cut Material
    c. Body: Wing
    d. Finished:
    e. 

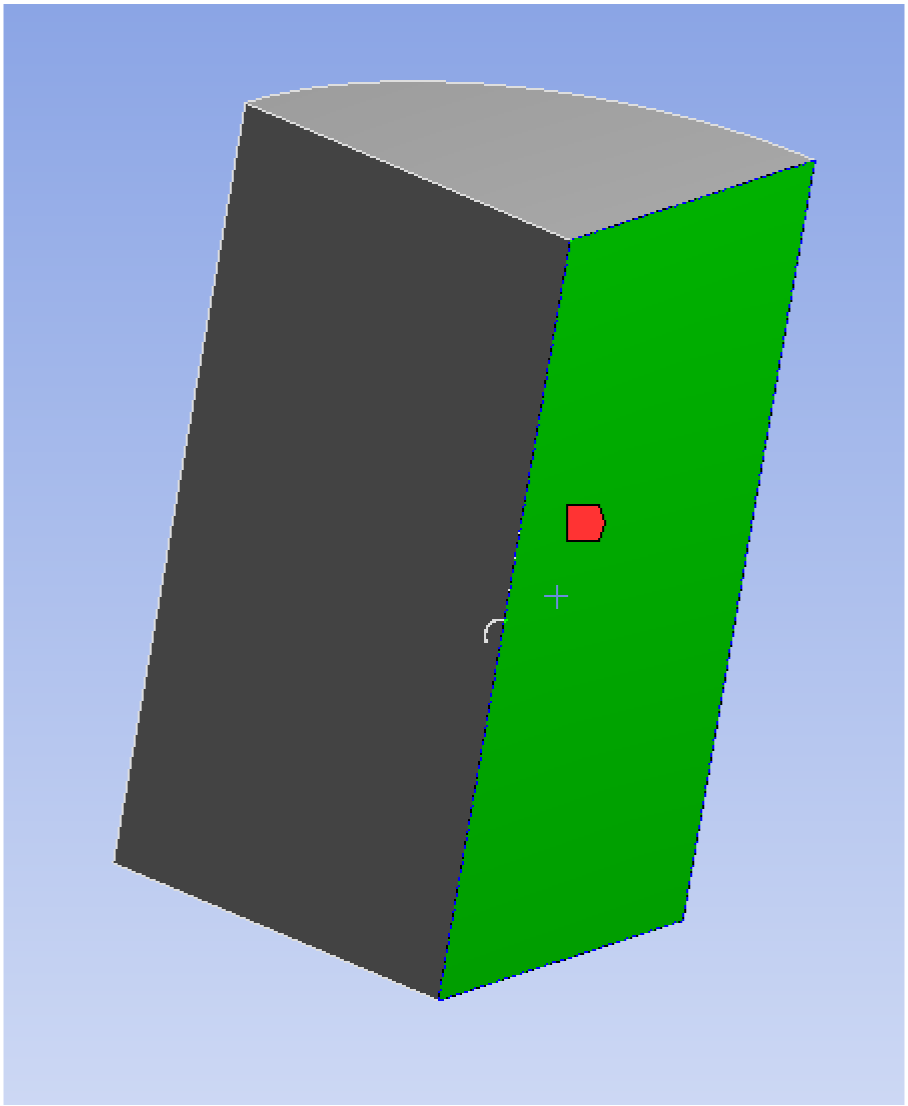
Section K.2 Meshing Part:



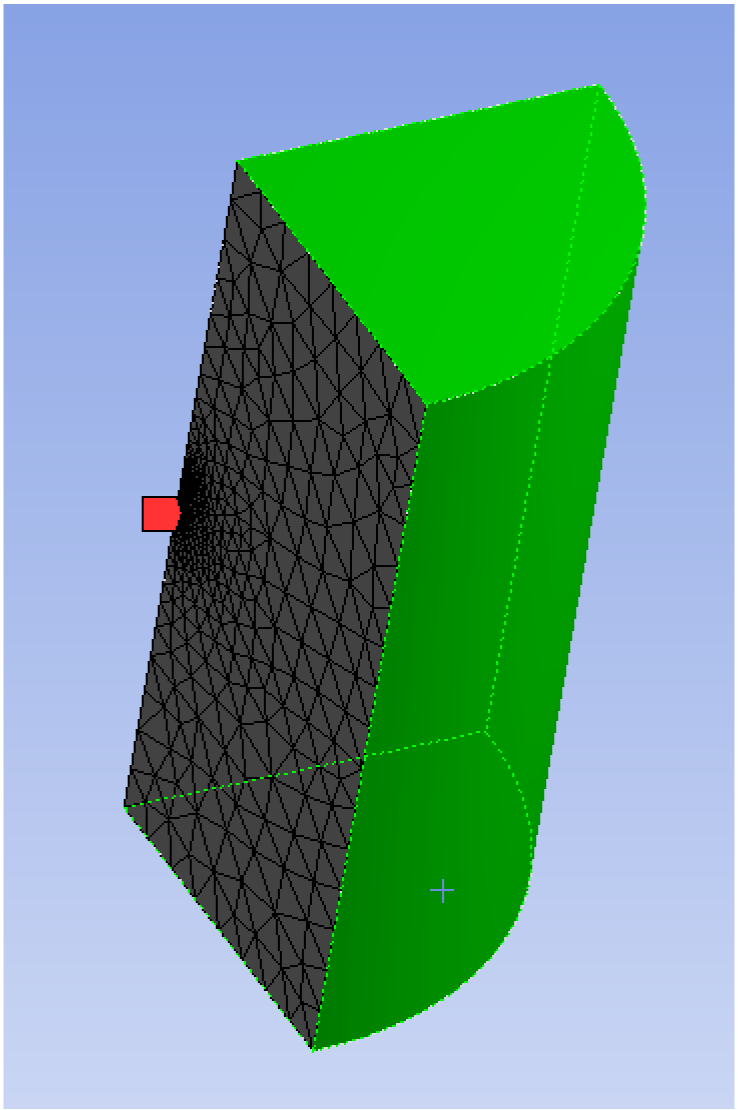
	A	
1	Fluid Flow (CFX)	
2	Geometry	✓
3	Mesh	⚡
4	Setup	?
5	Solution	?
6	Results	?

Fluid Flow (CFX)

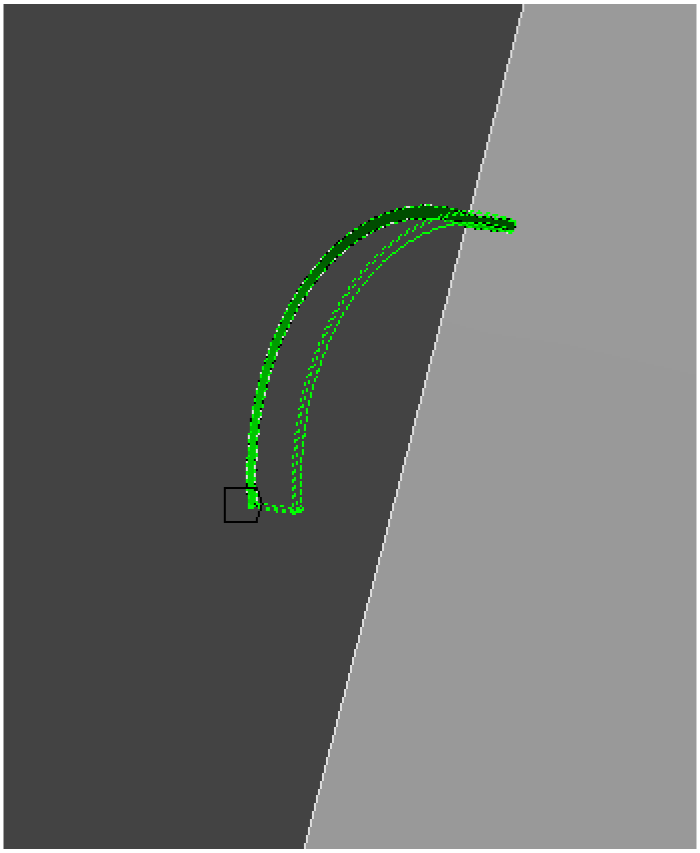
1. Generate a mesh based on CFD Physics preference and CFX solver preference.
2. Make named selections SymX, SymY, Opening, WingBulk
   a. Right click on Model
   b. Insert > Named Selection
   c. SymX: symmetry plane perpendicular to x
   d. SymY: symmetry plane perpendicular to y

e. 

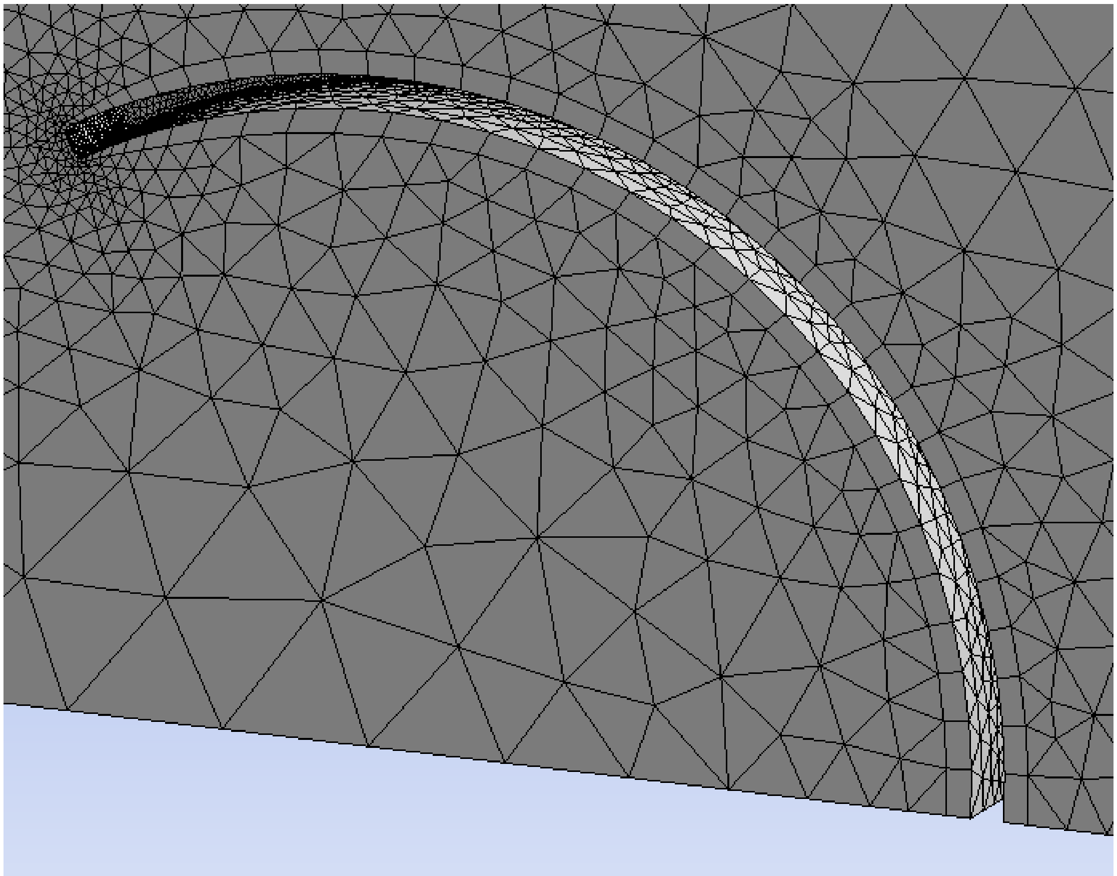
f. Opening: the other 3 edges of the cylindrical boundary region

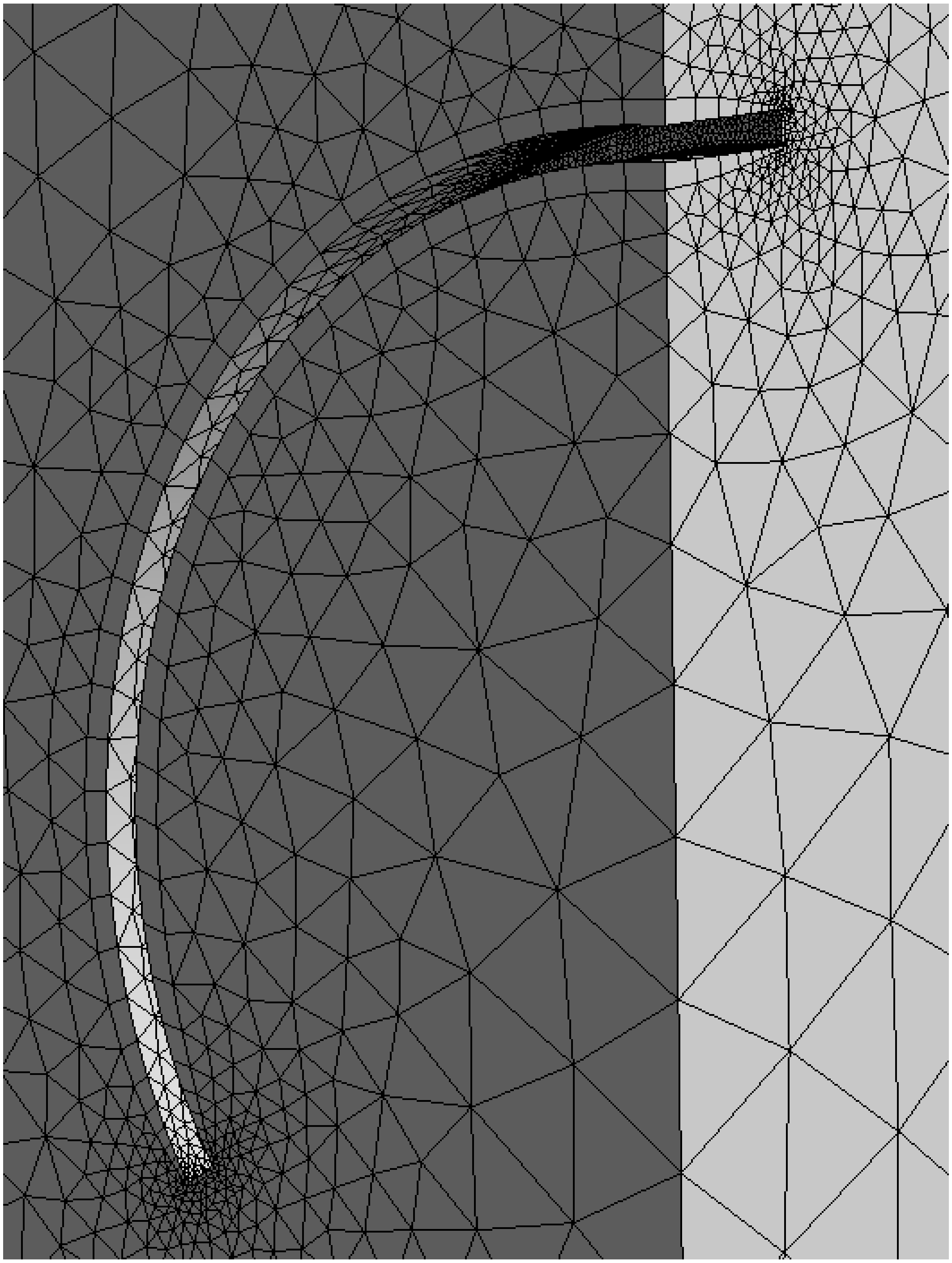
g. 

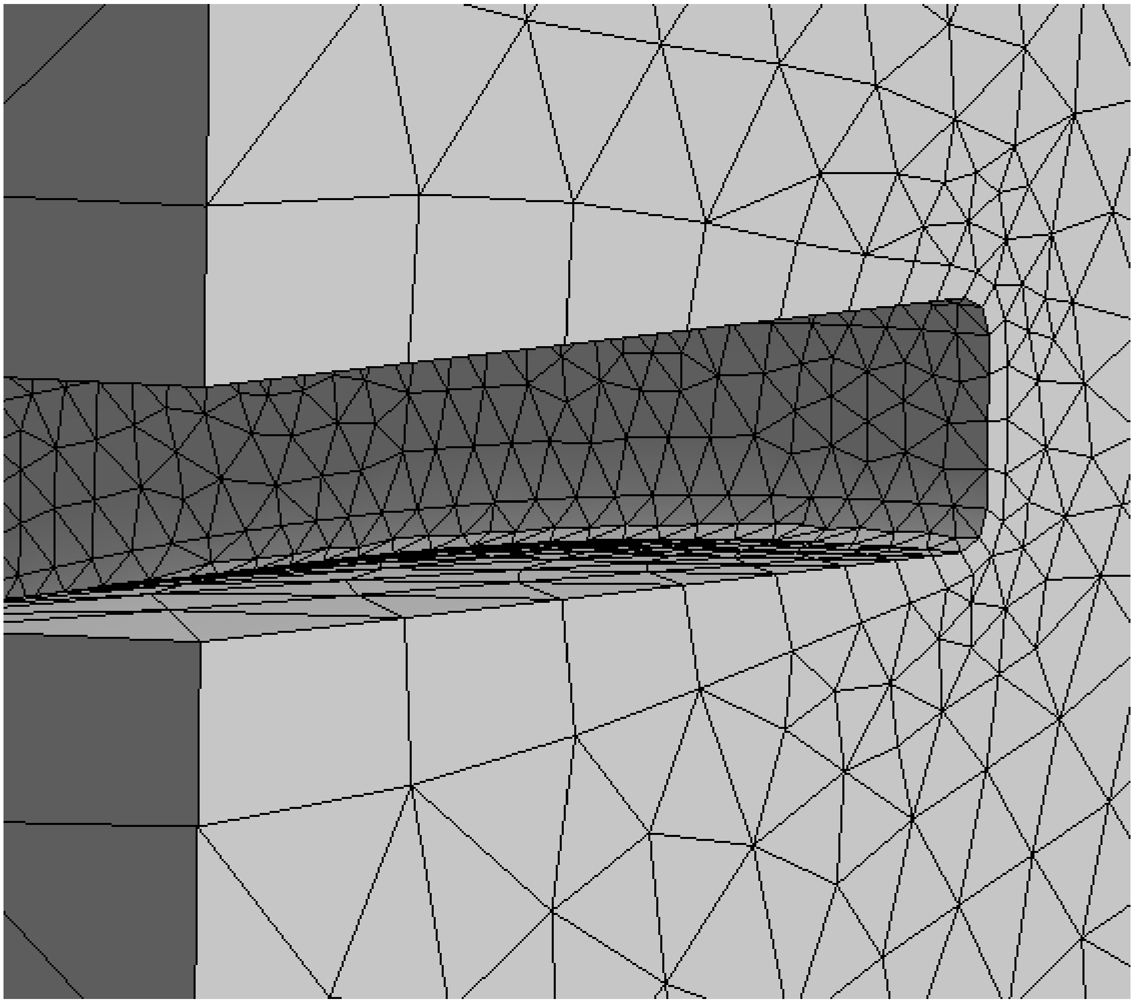
h. WingBulk: Remaining sections; all sides of the wing (11 faces)

i. 

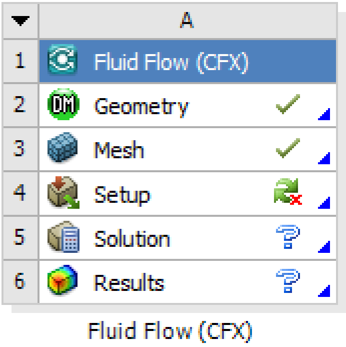
3. Set WingBulk's option to "Include" in "Program Controlled Inflation"
4. Mesh Properties
    a. Change sizing to: Min size = 8e-6 m, max face size = 2e-3 m, max size = default; growth rate = 1.25.
    b. Inflation >
        i. Use Automatic Inflation > Program Controlled
        ii. Maximum Layers > 1
        iii. Growth Rate > 1.1
    c. elements: 165854







Section K.3 Creating the Solver Definition File in CFX-Pre:

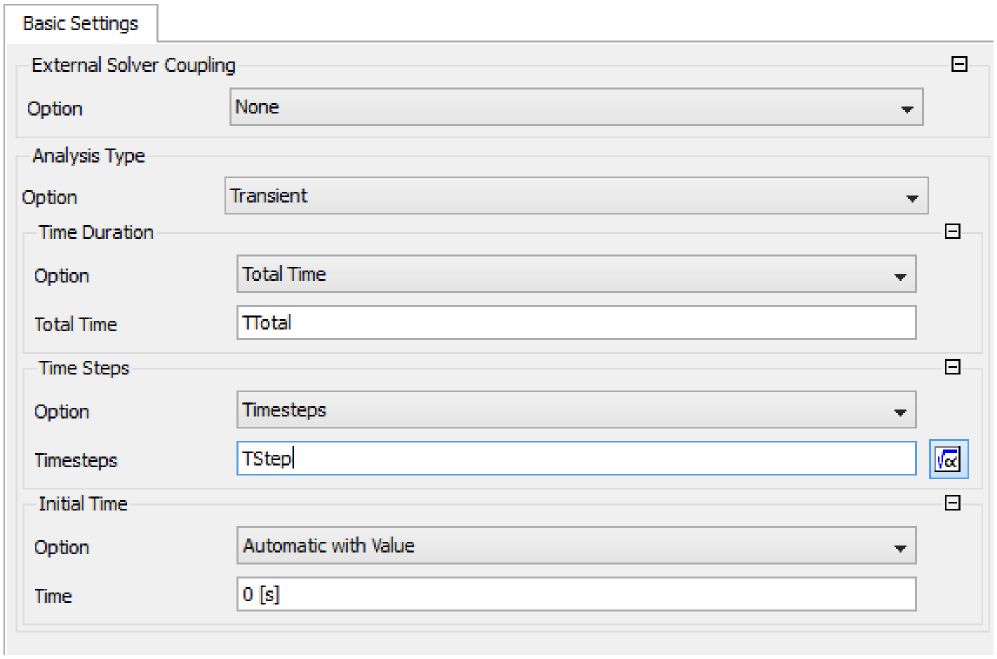


	A	
1	Fluid Flow (CFX)	
2	Geometry	✓
3	Mesh	✓
4	Setup	
5	Solution	?
6	Results	?

Fluid Flow (CFX)

Double click "Setup" to launch CFX-Pre. This program is where you use the mesh file to specify the simulation that the CFX-Solver will carry out.

1. Create Boundaries
    a. Right Click on "Default Domain" and insert 4 boundaries with the same names as the mesh sections: SymX, SymY, Opening, WingBulk
    b. SymX
        i. Boundary Type = Symmetry
        ii. Location = SymX
    c. SymY
        i. Boundary Type = Symmetry
        ii. Location = SymY
    d. Opening
        i. Boundary Type = Opening
        ii. Location = Opening
        iii. Relative Pressure = 0
    e. WingBulk
        i. Boundary Type = Wall
        ii. Location = WingBulk
2. Analysis Type: Transient
    a. Total Time = TotalTime
    b. Timesteps = Timestep

c. 

Basic Settings

External Solver Coupling

Option: None

Analysis Type

Option: Transient

Time Duration

Option: Total Time

Total Time: TTotal

Time Steps

Option: Timesteps

Timesteps: TStep

Initial Time

Option: Automatic with Value

Time: 0 [s]

3. Input the CCL code:
   a. Create any temporary expression, then
   b. Right click on it and select "Edit in Command Editor"
   c. Replace the text with this CCL code (explained in the next section):

```
LIBRARY:
  CEL:
    &replace EXPRESSIONS:
      Ar = Ar0*sin(6.28318530718*atstep/TSteps)
      Ar0 = 0.1
      AreaDirect = area()@WingBulk
      B = 0.5e-3[m]
      Clam = (cos(lam)+cosh(lam))/(sin(lam)+sinh(lam))
      Dx = Dx0*L+Dxz
      Dx0 = cx1*S^1+cx2*S^2+cx3*S^3+cx4*S^4+cx5*S^5
      Dxz = -z0*(sin(beta1)-sin(beta0))
      Dy = Dy0*L+Dyz
      Dy0 = cy1*S^1+cy2*S^2+cy3*S^3+cy4*S^4+cy5*S^5
      Dyz = z0*(cos(beta1)-cos(beta0))
      Fz = force_z()@WingBulk
      L = 2e-3[m]
      PowerUsed = -areaInt(dPbydA)@WingBulk
      S = s/L
      TCycle = TSteps*Timestep
      TSteps = 344
      Timestep = 8.44589062485e-07[s]
      TotalTime = 12*TCycle
      beta0 = S*phi0rad
      beta0g = S*phi0grad
      beta1 = S*phi0rad +Ar/2.0*lam*(sinh(slam)+sin(slam)+Clam*(cos(slam)-cosh(slam)))
      c0 = phi0/L
      c0g = phi0g/L
      cx1 = -1.83190599725e-06+0.0114197280605*Ar+0.00955589778247*Ar^2-
0.0228687858247*Ar^3+0.00930306144494*Ar^4+0.00127266550374*Ar^5
      cx2 = 2.12982851148e-05-0.158155863426*Ar-0.0794460144141*Ar^2+0.268143472532*Ar^3-
0.12770357418*Ar^4-0.00948678484208*Ar^5
      cx3 = -7.18547358598e-05+3.25349903498*Ar-2.08584155058*Ar^2-
0.877567935534*Ar^3+0.537596966458*Ar^4+0.000659796933706*Ar^5
      cx4 = 7.36971404931e-05-3.0799256064*Ar+3.48019344616*Ar^2+0.30137510422*Ar^3-
0.63522265112*Ar^4+0.0801943771275*Ar^5
      cx5 = -2.11905534952e-05+0.809669119156*Ar-
1.41807612032*Ar^2+0.115955345299*Ar^3+0.220105311812*Ar^4-0.0553806900559*Ar^5
      cy1 = 3.04619819663e-07-0.0115170123179*Ar+0.0399779683927*Ar^2-
0.0163457846018*Ar^3-0.00680178513738*Ar^4+0.00302678698426*Ar^5
      cy2 = 4.52922406367e-07+1.93378522823*Ar-
0.54902649457*Ar^2+0.269817013498*Ar^3+0.0684537253067*Ar^4-0.0376090365062*Ar^5
      cy3 = -2.37199541578e-05-1.53648615992*Ar+2.5395346435*Ar^2-1.52701484589*Ar^3-
0.157370633995*Ar^4+0.138253596144*Ar^5
      cy4 = 7.72944327477e-05-0.77767965679*Ar-1.80278524119*Ar^2+1.98479340019*Ar^3-
0.0895552722834*Ar^4-0.15192056819*Ar^5
      cy5 = -4.45363391066e-05+0.64226306852*Ar+0.284586904661*Ar^2-
0.735008291995*Ar^3+0.119008616212*Ar^4+0.0486130520136*Ar^5
      delY = Dx+delY0g*(1-exp(-t/(Timestep*2)))/2
      delY0g = (1/c0-z0)*(sin(s*c0)) - (1/c0g-z0)*(sin(s*c0g))
      delZ = Dy+delZ0g*(1-exp(-t/(Timestep*2)))/2
      delZ0g = (1/c0-z0)*(1-cos(s*c0)) - (1/c0g-z0)*(1-cos(s*c0g))
      lam = 1.8751
      mu = 1.831000000E-05
      phi0 = -120[deg]
      phi0g = -120[deg]
      phi0grad = phi0g*3.14159[rad]/180[deg]
      phi0rad = phi0*3.14159[rad]/180[deg]
      s = 1/c0g*atan2(c0g*(y-Total Mesh Displacement Y),(1-c0g*(z-Total Mesh Displacement
Z)))
      slam = S*lam
      z0 = -sin(beta0g)*(y-Total Mesh Displacement Y) + (z-Total Mesh Displacement
Z)*cos(beta0g)+(1-cos(beta0g))/c0g
    END
  END
END
```

   d. Click "Process" & "Close".

4. Right Click "Additional Variables" and "Insert Additional Variable"
   a. Name = dPbydA
   b. Units = [kg/s^3]
5. Edit Domain: Default Domain
   a. Mesh Deformation:
      i. Option = Regions of Motion Specified
      ii. Mesh Stiffness
         1. Option = Value
         2. Mesh Stiffness = (1 [m^3.5/s]/(Wall Distance+1e-12[m]))/sqrt(Wall Distance+1e-12[m])
   b. Fluid Models
      i. Heat Transfer:
         1. Option = None
      ii. Turbulence:
         1. Option = Shear Stress Transport
      iii. Additional Variable Models
         1. Additional Variable
            a. dPbydA: Check as active
            b. Option = Algebraic Equation
            c. Add. Var. Value = (Wall Shear X + Pressure*Normal X)*Mesh Velocity X + (Wall Shear Y + Pressure*Normal Y)*Mesh Velocity Y + (Wall Shear Z + Pressure*Normal Z)*Mesh Velocity Z
   c. Initialization:
      i. Check: Domain Initialization
      ii. Initial Conditions: Velocity Type = Cartesian
      iii. Cartesian Velocity Components: Option = Automatic with Value
         1. U = 0

2. V = 0

3. W = 0

iv. Static Pressure: Option = Automatic with Value

1. Relative pressure = 0

v. Turbulence: Option = Medium (Intensity = 5%)

6. Boundary Conditions, part 2: Revisit to set the mesh motion and the Turbulence boundary conditions now that they're set up.

   a. Opening

      i. Mesh Motion = stationary

   b. SymX:

      i. Mesh Motion = unspecified

   c. SymY:

      i. Mesh Motion = unspecified

   d. WingBulk:

      i. Mesh Motion:

         1. Option = Specified Displacement

         2. X Component = 0

         3. Y Component = delY

         4. Z Component = delZ

7. Solver Units

   a. Length Units = [mm] (this is necessary to prevent certain round-off errors for with the small length values in the mesh stiffness variable)

8. Solver Control: Set up Solver to search for solutions to higher accuracy due to the sensitivity of the force measurement.

   a. Basic Settings:

      i. Convergence Criteria

         1. Residual Target = 1e-5

   b. Equation Class Settings

i. Mesh Displacement Set to help prevent stability problems from an improperly deformed wing.

1. Convergence Control

a. Max. Coeff. Loops = 50

b. Min. Coeff. Loops = 1

2. Convergence Criteria

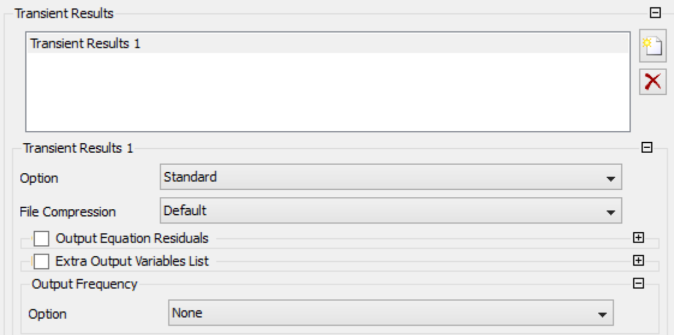
a. Residual Type = RMS

b. Residual Target = 1e-5

9. Output Control:

a. Trn Results Prevent the simulation from outputting results at every time step because these are hard-drive intensive and not necessary unless making a video of the flow.

i.



Transient Results

Transient Results 1

Transient Results 1

Option: Standard

File Compression: Default

□ Output Equation Residuals

□ Extra Output Variables List

Output Frequency

Option: None

ii. New

iii. Output Frequency = None

b. Monitor Set up the numeric outputs of the parameters that we actually need from the sims.

i. Monitor Objects

1. Monitor Points and Expressions

a. AreaMon

i. Option = Expression

ii. Expression Value = AreaDirect

b. FzMon

i. Option = Expression

        ii. Expression Value = Fz

    c. PowerUsedMon

        i. Option = Expression

        ii. Expression Value = PowerUsed

10. Simulation Control:

    a. Insert: Execution Control

        i. Executable Selection = Double Precision

Next, we need to save the definition from the CFX file in a format that's usable for batch mode processing on sharcnet. Here's how:

1. Close Workbench
2. Navigate to the folder where you saved the Workbench project, say, "phi0g120"
3. Find the CFX file in the directory tree; likely in:
   `phi0g120\phi0g120_files\dp0\CFX\CFX`
4. Open this file in CFX-Pre in standalone mode (i.e., not through Workbench).
5. *Now there are additional options*
6. File > Save Case As:

    a. `RS166kf0g120DP_R5_mm_SST.cfx`

        i. Deciphering this:

        ii. RS: Resonant shape

        iii. 166k: 166000 mesh elements

        iv. f0g120: ϕ_0 for the geometry = 120^o

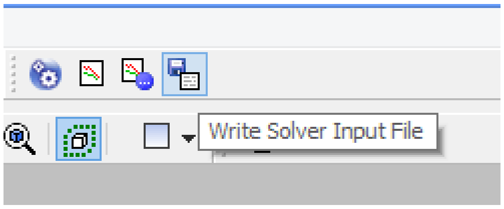
        v. DP: Double precision

        vi. R5: Residual convergence criteria = 1e-5

        vii. mm: Length scale = mm

        viii. SST: Turbulence model = Shear stress transport

7. Use the toolbar button "Write Solver Input File":

a. 

b. Rename as `RS166kf0g120DP_R5_mm_SST.def`

The .def file is the one the sharcnet solver actually needs to begin a simulation.

Section K.4 Generating the RS Fits for 120°

In the previous section on Creating the Solver Definition File in CFX-Pre, we needed a CCL input file to define the variables. Here's how to generate that:

Run RSVelocityWork.py (the program from Appendix J) with this in the main_loop():

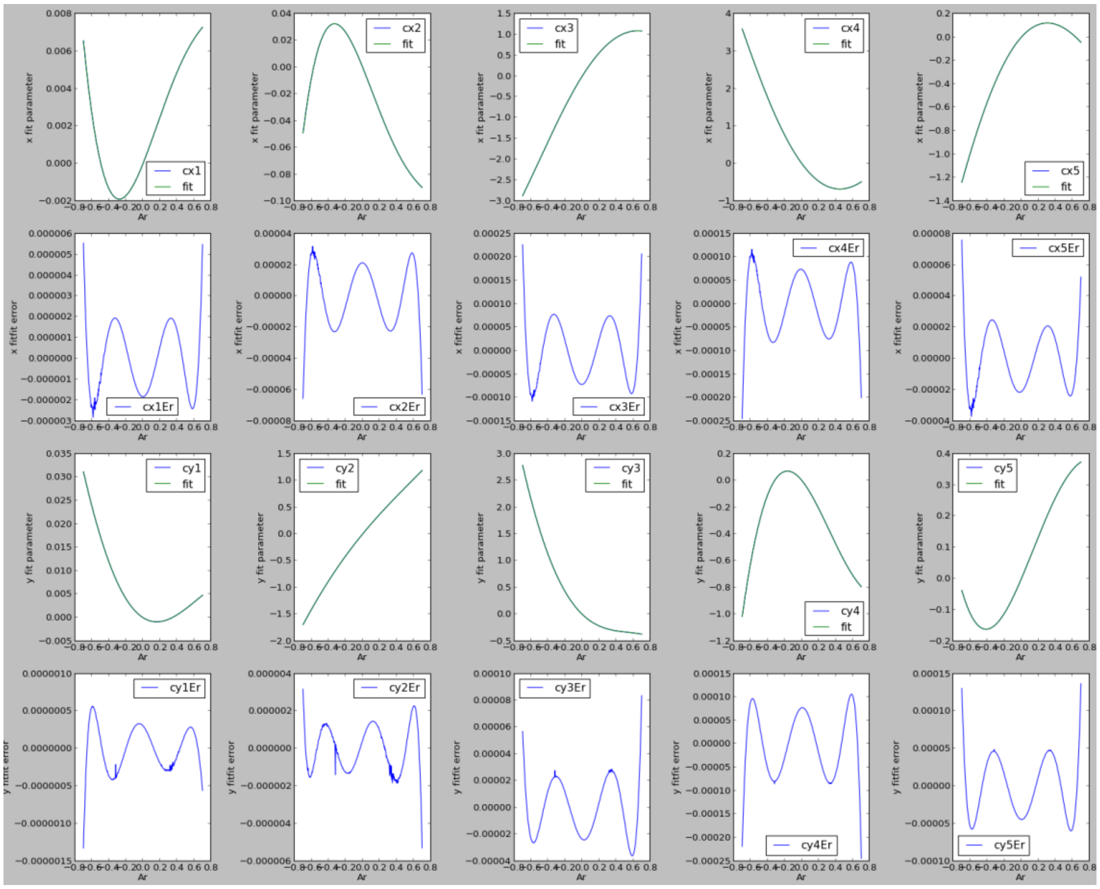
```
    phi0=-120.
    derive_fits_vs_Ar(phi0)
```

This produces the output (after 30 s on an AMD FX 8 Core Black Edition machine circa April 2013):

```
----------fit starting
cx1 = -1.83190599725e-06+0.0114197280605*Ar+0.00955589778247*Ar**2-
0.0228687858247*Ar**3+0.00930306144494*Ar**4+0.00127266550374*Ar**5
cx2 = 2.12982851148e-05-0.158155863426*Ar-
0.0794460144141*Ar**2+0.268143472532*Ar**3-0.12770357418*Ar**4-
0.00948678484208*Ar**5
cx3 = -7.18547358598e-05+3.25349903498*Ar-2.08584155058*Ar**2-
0.877567935534*Ar**3+0.537596966458*Ar**4+0.000659796933706*Ar**5
cx4 = 7.36971404931e-05-
3.0799256064*Ar+3.48019344616*Ar**2+0.30137510422*Ar**3-
0.63522265112*Ar**4+0.0801943771275*Ar**5
cx5 = -2.11905534952e-05+0.809669119156*Ar-
1.41807612032*Ar**2+0.115955345299*Ar**3+0.220105311812*Ar**4-
0.0553806900559*Ar**5
```

```
cy1 = 3.04619819663e-07-0.0115170123179*Ar+0.0399779683927*Ar**2-
0.0163457846018*Ar**3-0.00680178513738*Ar**4+0.00302678698426*Ar**5
cy2 = 4.52922406367e-07+1.93378522823*Ar-
0.54902649457*Ar**2+0.269817013498*Ar**3+0.0684537253067*Ar**4-
0.0376090365062*Ar**5
cy3 = -2.37199541578e-05-1.53648615992*Ar+2.5395346435*Ar**2-
1.52701484589*Ar**3-0.157370633995*Ar**4+0.138253596144*Ar**5
cy4 = 7.72944327477e-05-0.77767965679*Ar-
1.80278524119*Ar**2+1.98479340019*Ar**3-0.0895552722834*Ar**4-
0.15192056819*Ar**5
cy5 = -4.45363391066e-05+0.64226306852*Ar+0.284586904661*Ar**2-
0.735008291995*Ar**3+0.119008616212*Ar**4+0.0486130520136*Ar**5
Dx =cx1*S**1+cx2*S**2+cx3*S**3+cx4*S**4+cx5*S**5
Dy =cy1*S**1+cy2*S**2+cy3*S**3+cy4*S**4+cy5*S**5
```

As well as these plots detailing the fit goodness:



Appendix Figure K-1: Plots detailing the fit goodness for 120°

Note that the error is relatively very small (the blue plots detailing error in the fit have much smaller magnitudes than the green ones detailing the absolute value of the result of it).

The text output is formatted for Python (where ** means exponentiation). To place it into CFX, we need to replace all the **'s with ^'s (using a find and replace in say, Notepad++):

```
----------fit starting
cx1 = -1.83190599725e-06+0.0114197280605*Ar+0.00955589778247*Ar^2-
0.0228687858247*Ar^3+0.00930306144494*Ar^4+0.00127266550374*Ar^5
cx2 = 2.12982851148e-05-0.158155863426*Ar-
0.0794460144141*Ar^2+0.268143472532*Ar^3-0.12770357418*Ar^4-
0.00948678484208*Ar^5
```

```
cx3 = -7.18547358598e-05+3.25349903498*Ar-2.08584155058*Ar^2-
0.877567935534*Ar^3+0.537596966458*Ar^4+0.000659796933706*Ar^5
cx4 = 7.36971404931e-05-
3.0799256064*Ar+3.48019344616*Ar^2+0.30137510422*Ar^3-
0.63522265112*Ar^4+0.0801943771275*Ar^5
cx5 = -2.11905534952e-05+0.809669119156*Ar-
1.41807612032*Ar^2+0.115955345299*Ar^3+0.220105311812*Ar^4-
0.0553806900559*Ar^5
cy1 = 3.04619819663e-07-0.0115170123179*Ar+0.0399779683927*Ar^2-
0.016345784601 8*Ar^3-0.00680178513738*Ar^4+0.00302678698426*Ar^5
cy2 = 4.52922406367e-07+1.93378522823*Ar-
0.54902649457*Ar^2+0.269817013498*Ar^3+0.0684537253067*Ar^4-
0.0376090365062*Ar^5
cy3 = -2.37199541578e-05-1.53648615992*Ar+2.5395346435*Ar^2-
1.52701484589*Ar^3-0.157370633995*Ar^4+0.138253596144*Ar^5
cy4 = 7.72944327477e-05-0.77767965679*Ar-
1.80278524119*Ar^2+1.98479340019*Ar^3-0.0895552722834*Ar^4-
0.15192056819*Ar^5
cy5 = -4.45363391066e-05+0.64226306852*Ar+0.284586904661*Ar^2-
0.735008291995*Ar^3+0.119008616212*Ar^4+0.0486130520136*Ar^5
Dx =cx1*S^1+cx2*S^2+cx3*S^3+cx4*S^4+cx5*S^5
Dy =cy1*S^1+cy2*S^2+cy3*S^3+cy4*S^4+cy5*S^5
```

Finally, paste into the CFX file, after modifying the names of Dx & Dy to Dx0 & Dy0, respectively:

```
            elif Base[:12] == 'RS166kf0g120':
                L='2e-3[m]'
                phi0g = '-120[deg]'

                CCLString = '''LIBRARY:
      CEL:
        &replace EXPRESSIONS:
          Ar = Ar0*sin(omega*tn)
          Ar0 = {Ar}
          AreaDirect = area()@WingBulk
          B = 0.5e-3[m]
          Clam = (cos(lam)+cosh(lam))/(sin(lam)+sinh(lam))
          Dx = Dx0*L+Dxz
          Dx0 = cx1*S^1+cx2*S^2+cx3*S^3+cx4*S^4+cx5*S^5
          Dxz = -z0*(sin(beta1)-sin(beta0))
          Dy = Dy0*L+Dyz
          Dy0 = cy1*S^1+cy2*S^2+cy3*S^3+cy4*S^4+cy5*S^5
          Dyz = z0*(cos(beta1)-cos(beta0))
          Fz = force_z()@WingBulk
          L = {L}
          PowerUsed = -areaInt(dPbydA)@WingBulk
          S = s/L
          TCycle = 1/(omega/2/3.14159)
          TStep = TCycle/{TCycle}
          TTotal = {TTotal}*TCycle
          beta0 = S*phi0rad
      beta0g = S*phi0grad
          beta1 = S*phi0rad
+Ar/2.0*lam*(sinh(slam)+sin(slam)+Clam*(cos(slam)-cosh(slam)))
          c0 = phi0/L
          c0g = phi0g/L

        cx1 = -1.83190599725e-
06+0.0114197280605*Ar+0.00955589778247*Ar^2-
0.0228687858247*Ar^3+0.00930306144494*Ar^4+0.00127266550374*Ar^5
        cx2 = 2.12982851148e-05-0.158155863426*Ar-
0.0794460144141*Ar^2+0.268143472532*Ar^3-0.12770357418*Ar^4-
0.00948678484208*Ar^5
        cx3 = -7.18547358598e-05+3.25349903498*Ar-2.08584155058*Ar^2-
0.877567935534*Ar^3+0.537596966458*Ar^4+0.000659796933706*Ar^5
        cx4 = 7.36971404931e-05-
3.0799256064*Ar+3.48019344616*Ar^2+0.30137510422*Ar^3-
0.63522265112*Ar^4+0.0801943771275*Ar^5
        cx5 = -2.11905534952e-05+0.809669119156*Ar-
1.41807612032*Ar^2+0.115955345299*Ar^3+0.220105311812*Ar^4-
0.0553806900559*Ar^5
```

```
        cy1 = 3.04619819663e-07-0.0115170123179*Ar+0.0399779683927*Ar^2-
0.0163457846018*Ar^3-0.00680178513738*Ar^4+0.00302678698426*Ar^5
        cy2 = 4.52922406367e-07+1.93378522823*Ar-
0.54902649457*Ar^2+0.269817013498*Ar^3+0.0684537253067*Ar^4-
0.0376090365062*Ar^5
        cy3 = -2.37199541578e-05-1.53648615992*Ar+2.5395346435*Ar^2-
1.52701484589*Ar^3-0.157370633995*Ar^4+0.138253596144*Ar^5
        cy4 = 7.72944327477e-05-0.77767965679*Ar-
1.80278524119*Ar^2+1.98479340019*Ar^3-0.0895552722834*Ar^4-
0.15192056819*Ar^5
        cy5 = -4.45363391066e-05+0.64226306852*Ar+0.284586904661*Ar^2-
0.735008291995*Ar^3+0.119008616212*Ar^4+0.0486130520136*Ar^5
        Dx0 =cx1*S^1+cx2*S^2+cx3*S^3+cx4*S^4+cx5*S^5
        Dy0 =cy1*S^1+cy2*S^2+cy3*S^3+cy4*S^4+cy5*S^5

          delY = Dx+delY0g*(1-exp(-tn/(TStep*2)))/2
          delY0g = (1/c0-z0)*(sin(s*c0)) - (1/c0g-z0)*(sin(s*c0g))
          delZ = Dy+delZ0g*(1-exp(-tn/(TStep*2)))/2
          delZ0g = (1/c0-z0)*(1-cos(s*c0)) - (1/c0g-z0)*(1-cos(s*c0g))
          f = {f}[s^-1]
          lam = 1.8751
          omega = 2*3.14159*f
          phi0 = -{phi0}[deg]
          phi0g = {phi0g}
          phi0grad = phi0g*3.14159[rad]/180[deg]
          phi0rad = phi0*3.14159[rad]/180[deg]
          s = 1/c0g*atan2(c0g*(y-Total Mesh Displacement Y),(1-c0g*(z-
Total Mesh Displacement Z)))
          slam = S*lam
          tn = t+TStep/8
          tnG = tn*(1-exp(-tn/(TStep*4)))
          tnp1 = t+9*TStep/8
          tnp1G = tnp1*(1-exp(-tnp1/(TStep*4)))
          z0 = -sin(beta0g)*(y-Total Mesh Displacement Y) + (z-Total
Mesh Displacement Z)*cos(beta0g)+(1-cos(beta0g))/c0g
        END
```

```
      END
    END'''.format(L=L, phi0g=phi0g, Ar=Ar, phi0=phi0, f=f,
TTotal=TTotal, TCycle=TCycle)
```

For example, with phi0 = 120, Ar = 0.2, ReSet = 8, TCycle = 344, and TTotal = 12, we should get ID00871_RS166kf0g120DP_R5_mm_SST_phi120_Ar0.2_Re2_T344_TT12.ccl:

```
LIBRARY:
      CEL:
        &replace EXPRESSIONS:
          Ar = Ar0*sin(omega*tn)
          Ar0 = 0.2
          AreaDirect = area()@WingBulk
          B = 0.5e-3[m]
          Clam = (cos(lam)+cosh(lam))/(sin(lam)+sinh(lam))
          Dx = Dx0*L+Dxz
          Dx0 = cx1*S^1+cx2*S^2+cx3*S^3+cx4*S^4+cx5*S^5
          Dxz = -z0*(sin(beta1)-sin(beta0))
          Dy = Dy0*L+Dyz
          Dy0 = cy1*S^1+cy2*S^2+cy3*S^3+cy4*S^4+cy5*S^5
          Dyz = z0*(cos(beta1)-cos(beta0))
          Fz = force_z()@WingBulk
          L = 2e-3[m]
          PowerUsed = -areaInt(dPbydA)@WingBulk
          S = s/L
          TCycle = 1/(omega/2/3.14159)
          TStep = TCycle/344
          TTotal = 12*TCycle
          beta0 = S*phi0rad
      beta0g = S*phi0grad
          beta1 = S*phi0rad
+Ar/2.0*lam*(sinh(slam)+sin(slam)+Clam*(cos(slam)-cosh(slam)))
          c0 = phi0/L
          c0g = phi0g/L
        cx1 = -1.83190599725e-
06+0.0114197280605*Ar+0.00955589778247*Ar^2-
0.0228687858247*Ar^3+0.00930306144494*Ar^4+0.00127266550374*Ar^5
        cx2 = 2.12982851148e-05-0.158155863426*Ar-
0.0794460144141*Ar^2+0.268143472532*Ar^3-0.12770357418*Ar^4-
0.00948678484208*Ar^5
        cx3 = -7.18547358598e-05+3.25349903498*Ar-2.08584155058*Ar^2-
0.877567935534*Ar^3+0.537596966458*Ar^4+0.000659796933706*Ar^5
        cx4 = 7.36971404931e-05-
3.0799256064*Ar+3.48019344616*Ar^2+0.30137510422*Ar^3-
0.63522265112*Ar^4+0.0801943771275*Ar^5
        cx5 = -2.11905534952e-05+0.809669119156*Ar-
1.41807612032*Ar^2+0.115955345299*Ar^3+0.220105311812*Ar^4-
0.0553806900559*Ar^5
        cy1 = 3.04619819663e-07-0.0115170123179*Ar+0.0399779683927*Ar^2-
0.0163457846018*Ar^3-0.00680178513738*Ar^4+0.00302678698426*Ar^5
        cy2 = 4.52922406367e-07+1.93378522823*Ar-
0.54902649457*Ar^2+0.269817013498*Ar^3+0.0684537253067*Ar^4-
0.0376090365062*Ar^5
        cy3 = -2.37199541578e-05-1.53648615992*Ar+2.5395346435*Ar^2-
1.52701484589*Ar^3-0.157370633995*Ar^4+0.138253596144*Ar^5
        cy4 = 7.72944327477e-05-0.77767965679*Ar-
1.80278524119*Ar^2+1.98479340019*Ar^3-0.0895552722834*Ar^4-
0.15192056819*Ar^5
        cy5 = -4.45363391066e-05+0.64226306852*Ar+0.284586904661*Ar^2-
0.735008291995*Ar^3+0.119008616212*Ar^4+0.0486130520136*Ar^5
        Dx0 =cx1*S^1+cx2*S^2+cx3*S^3+cx4*S^4+cx5*S^5
        Dy0 =cy1*S^1+cy2*S^2+cy3*S^3+cy4*S^4+cy5*S^5

          delY = Dx+delY0g*(1-exp(-tn/(TStep*2)))/2
          delY0g = (1/c0-z0)*(sin(s*c0)) - (1/c0g-z0)*(sin(s*c0g))
          delZ = Dy+delZ0g*(1-exp(-tn/(TStep*2)))/2
          delZ0g = (1/c0-z0)*(1-cos(s*c0)) - (1/c0g-z0)*(1-cos(s*c0g))
          f = 2.732E+01[s^-1]
          lam = 1.8751
          omega = 2*3.14159*f
          phi0 = -120[deg]
          phi0g = -120[deg]
          phi0grad = phi0g*3.14159[rad]/180[deg]
          phi0rad = phi0*3.14159[rad]/180[deg]
          s = 1/c0g*atan2(c0g*(y-Total Mesh Displacement Y),(1-c0g*(z-
Total Mesh Displacement Z)))
          slam = S*lam
          tn = t+TStep/8
          tnG = tn*(1-exp(-tn/(TStep*4)))
          tnp1 = t+9*TStep/8
          tnp1G = tnp1*(1-exp(-tnp1/(TStep*4)))
          z0 = -sin(beta0g)*(y-Total Mesh Displacement Y) + (z-Total
Mesh Displacement Z)*cos(beta0g)+(1-cos(beta0g))/c0g
        END
      END
    END
```

Section K.5 Sharcnet Simulation Starting

These commands are subject to change as the sqsub system evolves, but this is correct at time of writing and likely similar to future commands.

1. Ensure that the proper .def file and .ccl files are in the working directory on sharcnet.
2. Load the ansys module:
   a. `module load ansys/14.0`
3. Load the commands to run simulations separated by semicolons into a shell script, say, ID00835.sh, and place that in the working directory as well.
   a. e.g., command to run a simulation:
   b.
   ```
   cfx RS166kf0g120DP_R5_mm_SST.def -fullname
   ID00871_RS166kf0g120DP_R5_mm_SST_phi120_Ar0.2_Re2_T344_TT12 -
   ccl
   ID00871_RS166kf0g120DP_R5_mm_SST_phi120_Ar0.2_Re2_T344_TT12.ccl
   ```
4. Launch the shell file by submitting it via the sqsub command; e.g., on orca, for best results use the xeon cores:

a.
```
sqsub --idfile ID00835jid -r 7d --nompirun -q mpi -n 12 --
mpp=500M -f xeon -o ID00835.%J sh ID00835.sh
```

Appendix L Viscosity Scaling

Section L.1 Viscosity of nitrogen and air with temperature:

Viscosity vs. temperature from [67].

$$\mu = \mu_0 \frac{T_0 + C}{T + C}\left(\frac{T}{T_0}\right)$$

Or, $\mu = \lambda \frac{T^{3/2}}{T + C}$, where $\lambda = \frac{\mu_0 \left(T_0 + C\right)}{T_0^{3/2}}$ is a constant for each gas:

$$\lambda_{N_2} = 1.406732195, C_{N_2} = 111\mathrm{K}$$
$$\lambda_{Air} = 1.512041288, C_{Air} = 120\mathrm{K}$$

effect:

@ T = 19 degC = 292K,

$$\mu_{N_2} = 1.406732 \frac{\mu\text{Pa-s}}{\sqrt{\mathrm{K}}} \frac{\left(292\mathrm{K}\right)^{3/2}}{\left(292\mathrm{K}\right) + 111\mathrm{K}} = 17.417$$

$$\mu_{Air} = 1.512041 \frac{\mu\text{Pa-s}}{\sqrt{\mathrm{K}}} \frac{\left(292\mathrm{K}\right)^{3/2}}{\left(292\mathrm{K}\right) + 120\mathrm{K}} = 18.3122$$

and $\rho_{N2} = 1.165\mathrm{kg/m^3}$
$\rho_{Air} = 1.205\mathrm{kg/m^3}$

Section L.2 Full Detailed Viscosity vs. Temperature and Pressure

Based on work using [68],

Viscosity of a gas doesn't really depend on the pressure at all for partial vacuums:

1atm @ 300K: 17.890e-6 Pa-s

6.5%atm @ 300K: 17.878e-6 Pa-s

But depends a lot more on temperature:

@T = 293K, P = 1atm --> rho = 0.042mol/dm^3,

Total viscosity is eta = 17.565949585 microPa-s

($\eta \equiv \mu$, dynamic viscosity; remember that $\nu = \dfrac{\mu}{\rho}$:

$\mu = 17.57\text{e-}6$ Pa-s)

Appendix M Derivation of Basic Forced Response of a Linear System

Mass on a spring subject to a restoring force: $-kx$ and damping force $-b\dot{x}$ has the undriven (homogeneous) equation of motion:

$$m\ddot{x} = -b\dot{x} - kx$$

Assumed solution form:

$$x = e^{rt} \text{ -->}$$

$$\left(mr^2 + br + k\right)e^{rt} = 0$$

$$\rightarrow r = \frac{-b \pm \sqrt{b^2 - 4mk}}{2m}$$

For low damping relative to *mk*, $b^2 < 4mk$ and the square root is imaginary:

$$r = \frac{-b}{2m} \pm i\sqrt{\frac{k}{m} - \left(\frac{b}{2m}\right)^2}$$

giving the solutions: $x = e^{\frac{-b}{2m}t}\cos\left(\sqrt{\frac{k}{m} - \left(\frac{b}{2m}\right)^2}\, t\right), e^{\frac{-b}{2m}t}\sin\left(\sqrt{\frac{k}{m} - \left(\frac{b}{2m}\right)^2}\, t\right)$

Both of which describe ringdown of a response at the natural frequency,

$\omega_{0True} = \sqrt{\frac{k}{m} - \left(\frac{b}{2m}\right)^2}$. For sufficiently low damping, the frequency approaches $\sqrt{\frac{k}{m}}$.

Suppose the amplitude is *A* and the result is SHM. Then the peak energy stored for is $U = \tfrac{1}{2}kA^2$ (which equals $K = \tfrac{1}{2}mv_{\max}^2 = \tfrac{1}{2}m(\omega A)^2$) leading to a quality factor of $Q \equiv \frac{2\pi U}{TP}$, where $T = \frac{1}{f} = \frac{2\pi}{\omega}$ is the resonant period and $P = \int_0^T F_{damping} \cdot vdt$ is the average power going into damping over a cycle: $P = \int_0^T bv^2 dt = \int_0^T b\left(\omega A \sin \omega t\right)^2 dt = \tfrac{1}{2}b\omega^2 A^2$.

Therefore, $Q \equiv \frac{2\pi \tfrac{1}{2}kA^2}{\frac{2\pi}{\omega}\tfrac{1}{2}b\omega^2 A^2} = \frac{k}{b\omega}$. Substituting $k \equiv m\omega_0^2$, $Q = \frac{m\omega_0^2}{b\omega}$.

(Using the later definition of $b = \dfrac{\omega_0 m}{Q}$, where $\omega_0 = \sqrt{k/m}$, this gives a resonant frequency with non-ignorable damping of:

$$\omega_{0True} = \sqrt{\frac{k}{m} - \frac{k/m}{4Q^2}} = \omega_0\sqrt{1 - \frac{1}{4Q^2}}$$

The damped resonant frequency is ***lower***: $f_d = f_0\sqrt{1-\zeta^2}$, where $\zeta \equiv 1/(2Q)$ is the **damping ratio**.)

The general solution is then

$$x_c = e^{\frac{-b}{2m}t}\left(C_1 \cos\omega_0 t + C_2 \sin\omega_0 t\right)$$

Driving the oscillator with the signal $F = F_0 \sin(\omega t)$ leads to a system response obtainable by finding a particular solution to:

$$m\ddot{x} + b\dot{x} + kx = F_0 \sin\omega t$$

Try the method of undetermined coefficients; assume a particular solution of the form:

$$x_p = A\sin(\omega t) + B\cos\omega t$$

Then get the table:

Table 9-1: Derivatives for Method of Undetermined Coefficients for Linear System Forced Response DE

	$\sin\omega t$	$\cos\omega t$
x_p	A	B
$x_p{}'$	$-\omega B$	ωA
$x_p{}''$	$-\omega^2 A$	$-\omega^2 B$
$m\ddot{x} + b\dot{x} + kx$	$-m\omega^2 A - b\omega B + kA$	$-m\omega^2 B + b\omega A + kB$

F	F_0	*0*

Solve the equations: $\begin{cases} -m\omega^2 A - b\omega B + kA = F_0 \\ -m\omega^2 B + b\omega A + kB = 0 \end{cases}$

--> $A = \dfrac{m\omega^2 - k}{b\omega} B$

$$-\frac{\left(m\omega^2 - k\right)^2}{b\omega} B - b\omega B = F_0$$

--> $B = \dfrac{-F_0}{\dfrac{\left(m\omega^2 - k\right)^2}{b\omega} + b\omega} = \dfrac{-b\omega F_0}{\left(m\omega^2 - k\right)^2 + \left(b\omega\right)^2}$

$$\therefore A = \frac{-\left(m\omega^2 - k\right)F_0}{\left(m\omega^2 - k\right)^2 + \left(b\omega\right)^2}$$

And the particular solution found using the MUC is:

$$x_p = A\sin\omega t + B\cos\omega t$$

$$= \frac{-F_0}{\left(m\omega^2 - k\right)^2 + \left(b\omega\right)^2}\left[\left(m\omega^2 - k\right)\sin\omega t + b\omega\cos\omega t\right]$$

This sum of a sine and cos at the same frequency can be written as a single sine or cos with a phase constant by writing:

$$A\sin\alpha t + B\cos\alpha t = R\cos\left(\alpha t - \phi\right)$$

$$= R\cos\alpha t\cos\phi + R\sin\alpha t\sin\phi$$

$$\rightarrow \begin{cases} A = R\sin\phi \\ B = R\cos\phi \end{cases} \rightarrow \begin{cases} R = \sqrt{A^2 + B^2} \\ \phi = \tan^{-1}\left(A/B\right) \end{cases}$$

therefore,

$$x_p = \frac{-F_0}{\left(m\omega^2 - k\right)^2 + \left(b\omega\right)^2}\sqrt{\left(m\omega^2 - k\right)^2 + \left(b\omega\right)^2}\cos\left(\omega t - \tan^{-1}\left(\frac{m\omega^2 - k}{b\omega}\right)\right)$$

$$\therefore x_p = \frac{-F_0}{\sqrt{\left(m\omega^2 - k\right)^2 + \left(b\omega\right)^2}}\cos\left(\omega t - \tan^{-1}\left(\frac{m\omega^2 - k}{b\omega}\right)\right)$$

Rewrite in terms of $\omega_0 \equiv \sqrt{\frac{k}{m}}$:

$$x_p = \frac{-F_0}{m\sqrt{\left(\omega^2 - \frac{k}{m}\right)^2 + \left(\frac{b}{m}\omega\right)^2}} \cos\left(\omega t - \tan^{-1}\left(\frac{\omega^2 - k/m}{b\omega/m}\right)\right)$$

$$= \frac{-F_0/m}{\sqrt{\left(\omega^2 - \omega_0^2\right)^2 + \left(\frac{b}{m}\omega\right)^2}} \cos\left(\omega t - \tan^{-1}\left(\frac{\omega^2 - \omega_0^2}{b\omega/m}\right)\right)$$

At resonance, the amplitude is:

$\frac{F/m}{\left(\frac{b}{m}\omega_0\right)} = \frac{F}{b\sqrt{\frac{k}{m}}}$. Compared to the DC response, $x_{DC} \equiv F_0/k$, this is an amplification of

$$Q \equiv \left(\frac{1}{b\sqrt{\frac{k}{m}}}\right) / \frac{1}{k} = \frac{\sqrt{km}}{b} = \frac{\omega_0 m}{b}$$

Therefore, $x_p = \frac{-x_{DC}\omega_0^2}{\sqrt{\left(\omega^2 - \omega_0^2\right)^2 + \left(\frac{\omega_0\omega}{Q}\right)^2}} \cos\left(\omega t - \tan^{-1}\left(Q\frac{\omega^2 - \omega_0^2}{\omega_0\omega}\right)\right)$

As we move off-resonance, the amplitude drops as the denominator increases.

The response drops off to $1/\sqrt{2}$ of the resonant one at a frequency where

$$\omega^2 - \omega_0^2 = \frac{\omega_0}{Q}\omega \qquad (\text{for } \omega > \omega_0)$$

$$\omega^2 - \frac{\omega_0}{Q}\omega - \omega_0^2 = 0$$

$$\omega = \frac{\frac{\omega_0}{Q} \pm \sqrt{\left(\frac{\omega_0}{Q}\right)^2 + 4\omega_0^2}}{2}$$

$$\omega = \left(\frac{1}{Q} \pm \sqrt{\frac{1}{Q^2} + 4}\right)\frac{\omega_0}{2}$$

Need positive ω, so must have the positive root. For sufficiently large Q, $\sqrt{\left(\frac{1}{Q}\right)^2 + 4} \approx 2$,

so

$$\omega = \left(\frac{1}{Q} + 2\right)\frac{\omega_0}{2} = \frac{\omega_0}{2Q} + \omega_0$$

$$\therefore \omega - \omega_0 = \frac{\omega_0}{2Q}$$

$$\therefore Q = \frac{\omega_0}{2\Delta\omega}$$

where $\Delta\omega$ is the distance from resonance to the location where the amplitude drops by a factor of sqrt(2).

$$x_p = \frac{-x_{DC}}{\sqrt{\left(\left(\frac{\omega}{\omega_0}\right)^2 - 1\right)^2 + \left(\frac{\omega}{Q\omega_0}\right)^2}}\cos\left(\omega t - \tan^{-1}\left(Q\frac{\omega^2 - \omega_0^2}{\omega_0\omega}\right)\right)$$

$$x_p = \frac{-Qx_{DC}}{\sqrt{Q^2\left(\left(\frac{\omega}{\omega_0}\right)^2 - 1\right)^2 + \left(\frac{\omega}{\omega_0}\right)^2}}\cos\left(\omega t - \tan^{-1}\left(Q\frac{\omega^2 - \omega_0^2}{\omega_0\omega}\right)\right)$$

This determines the off-resonant response:

Section M.1 Off-resonance quality factor

(for the linear mass-spring derived in the previous section)

And the effective Q (i.e., amplitude amplification) for off-resonant actuation:

$$Q_{eff} = \frac{Q}{\sqrt{Q^2\left(\left(\frac{\omega}{\omega_0}\right)^2 - 1\right)^2 + \left(\frac{\omega}{\omega_0}\right)^2}}$$

Section M.2 Non-Ignorable Mechanical Damping

Mechanical damping in the linear regime gives a mechanical quality factor related to the mechanical power loss, P_{A4Mech}:

$$Q_{Mech} = \frac{2\pi U_{peak}}{P_{A4Mech} T}$$

The total quality factor would be:

$$Q_{True} = \frac{2\pi U_{peak}}{P_{ATotal} T} = \frac{2\pi U_{Peak}}{\left(P_{A4Mech} + P_{A4}\right) T} = \frac{1}{\frac{1}{Q_{Mech}} + \frac{1}{Q_{Fluidic}}}$$

Section M.3 Damped Resonance Frequency:

$$f_d = f_0 \sqrt{1 - \left(\frac{1}{2Q}\right)^2}$$

($f_d = f_0\sqrt{1-\zeta^2}$ where the damping ratio is $\zeta = \frac{1}{2Q}$, following [69])

For a mass-spring-linear damper, $m\ddot{x} + c\dot{x} + kx = 0$, the **damping ratio** is defined as $\zeta = \frac{c}{2\sqrt{km}}$. For forced vibrations, the amplitude of the vibration is:

$$X = \frac{F_0}{k}\frac{1}{\sqrt{\left(1-r^2\right)^2+\left(2\zeta r\right)^2}}$$

where *r* is the ratio of the forcing frequency to the *undamped* resonant frequency: $r \equiv \frac{f}{f_0}$.

Comparing with

$$Q_{eff} = \frac{Q}{\sqrt{Q^2\left(\left(\frac{\omega}{\omega_0}\right)^2-1\right)^2+\left(\frac{\omega}{\omega_0}\right)^2}}$$

we get that: $2\zeta r = \frac{\omega}{Q\omega_0} \rightarrow \zeta = \frac{1}{2Q}$.

Appendix N Material Properties for the Fabricated Wing

The fabricated wing's properties depend on several material properties: The resonant frequency is dependent on each layer's density and Young's modulus, the static bending angle depends on each layer's Young's modulus and built-in strain, the DC amplitude depends on each layer's Young's modulus and piezoelectric coupling coefficient, and the power drag coefficient's translation into a quality factor depends on each layer's density. This section compiles research on the numeric values of these properties.

Section N.1 Density

The density of each layer was retrieved from literature:

Quartz: 2651 kg/m^3 [COMSOL Materials Library: [70]]

SU-8: 1190 kg/m^3 [71]

Au: 19300 kg/m^3 [72]

Cr: 7190 kg/m^3 [72]

Section N.2 Young's Modulus & Poisson's Ratio

The Young's modulus was a fair bit trickier to obtain. The bridge at the end of the wings is very stiff in the width-direction, and so acts as a plane-strain boundary condition for the wings. This means that the effective Young's modulus for isotropic materials (like the metal layers and the SU-8) is related to the unbounded Young's modulus via

$$E_{effective} = \frac{E}{1-\nu^2}$$ [28].

Au: E = 78.0 GPa, $\nu = 0.44$ [72]

Cr: E = 279 GPa, $\nu = 0.21$ [73]

By carrying out simulations in COMSOL Multiphysics, I've verified that for quartz cantilevers with the crystal x-axis as the thickness direction and the crystal y-axis as the length direction, the effective Young's modulus with this boundary condition is 86.4 GPa. These simulations are explained in appendix 0.

Unfortunately SU-8’s Young’s modulus is highly variable, ranging from 2.5 to 4.95 GPa in literature ([74], [75]) (this is the raw *E*, before correction for 3d-boundary conditions). Because of this variation, we have less confidence in the exact Young’s modulus of SU-8 than we do in the DC-curvature formula, and so in the testing section, we’ll use that to calibrate the Young’s modulus of SU-8.

Section N.3 Built-in Strain

Built-in strain, like the Young’s modulus, is a particularly difficult to pin down parameter of SU-8. As Bachmann says in [76], "It has been observed that the material parameters of SU-8 strongly vary with slight variations of the process parameters. It is therefore not applicable to use literature values." Indeed, the built-in strain in deposited SU-8 in literature ranged even more than the Young’s modulus: from 4.6e-3 [77] to 11.1e-3 [76].

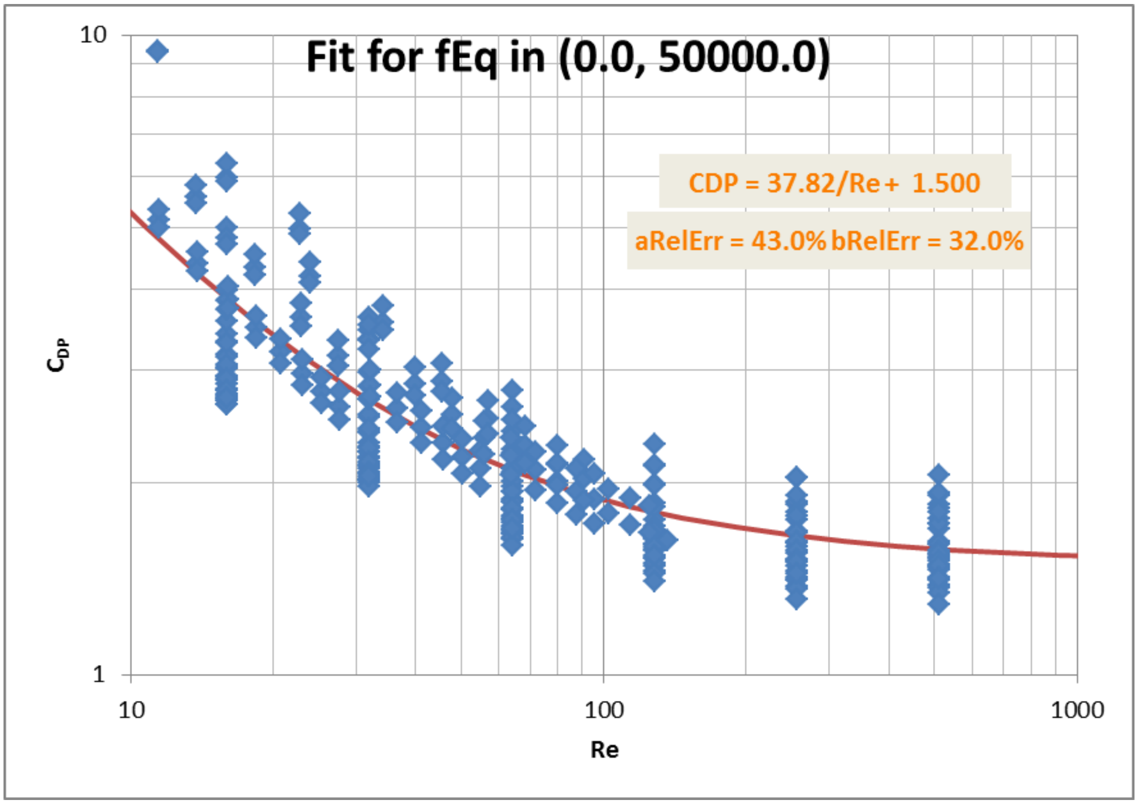
Section N.4 Piezoelectric constant

As mentioned in Section 2.2.6.6, the effective piezoelectric coefficient for quartz is modified due to the boundary conditions of the beam to be $d_{effective} = 106\% d_{21}$, where d_{21} = 2.3e-12 m/V [70], [78].

Appendix O Quality Factor Magnitude Calculations

First, we need to know what the power drag coefficient is.

Globally, the trend of the drag coefficient vs. Re is C_{DP} = 37.8/Re + 1.5. Really, drag coefficient depends on amplitude and offset as well, and even then this trend isn't always perfectly representative of the simulation, but even still this is within about a factor of 2 of the simulation result:



Appendix Figure O-1: Power drag coefficients vs. Re

Next, refer to the spreadsheet QSU8_17AngleMeasures_RSFix.xls which stores measured amplitudes measured for QSU8_17a. For each measurement pressure in turn (1, 0.7, 0.4, & 0.2 atm), we will:

1. Plot amplitude vs. frequency to find the resonance frequency at different voltages and times;
2. Plot CFX-determined drag coefficient for that resonant frequency at that pressure to obtain a local fit function for drag coefficient; then
3. Calculate the mechanical quality factor for each measurement near the resonant frequency and plot vs. applied voltage to look for trends.

Section 0.1 1 atm:

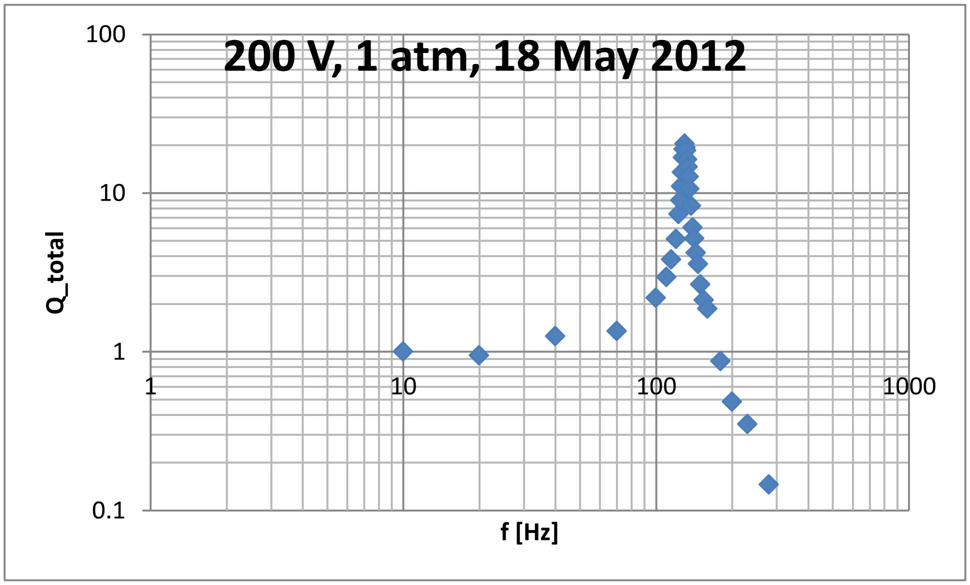
Section 0.1.1 18 May 200 V, 113-145, f10 to 280

Found a resonance at 130Hz

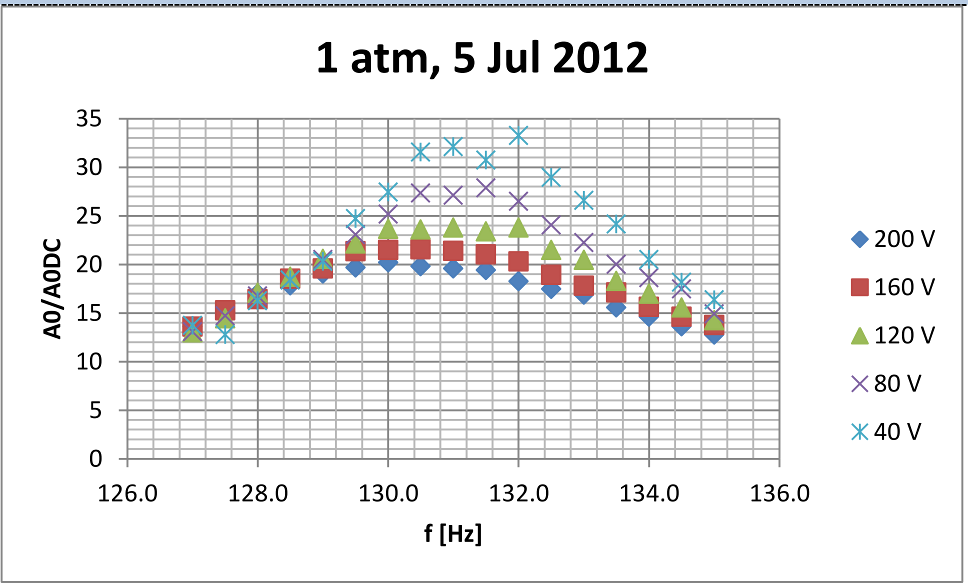
phi0 measure weirdly changes from 110^o to 117^o between 120 & 121.

V = 200, Ar0 = .1964

Qmech @ res = 39.6



Appendix Figure O-2: Total quality factor vs. frequency based on amplitude measurements at 200 V from 2012-05-18

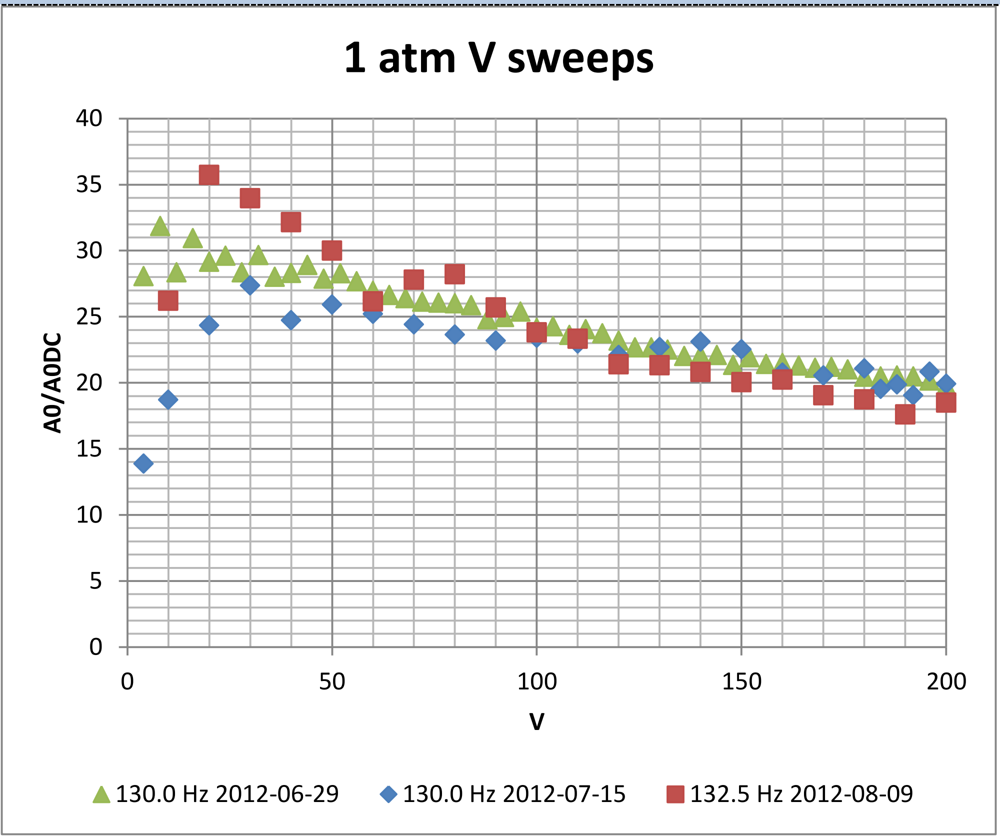


Appendix Figure O-3: Amplitude amplification measures vs. frequency at different voltages demonstrates the nonlinear frequency-dependent effect of resonance

This reveals a nonlinear nature to the resonant frequency. Resonant frequency for the pair of wings at 1 atm appears to be decreasing slightly with voltage, from around 131.5 Hz at 40 V to 130 Hz at 200 V.

This unfortunately means that sweeping voltage at a single frequency (i.e., 130 Hz) is not an effective way to measure the amplitude at resonance as a function of voltage. Rather, this approach likely underestimates the resonant amplitude at lower voltages: 0% at 200V, and 18% (27 vs. 33) at 40 V.

These sweeps had phi0 = 117^{o}, stdev = 0.4^{o}.

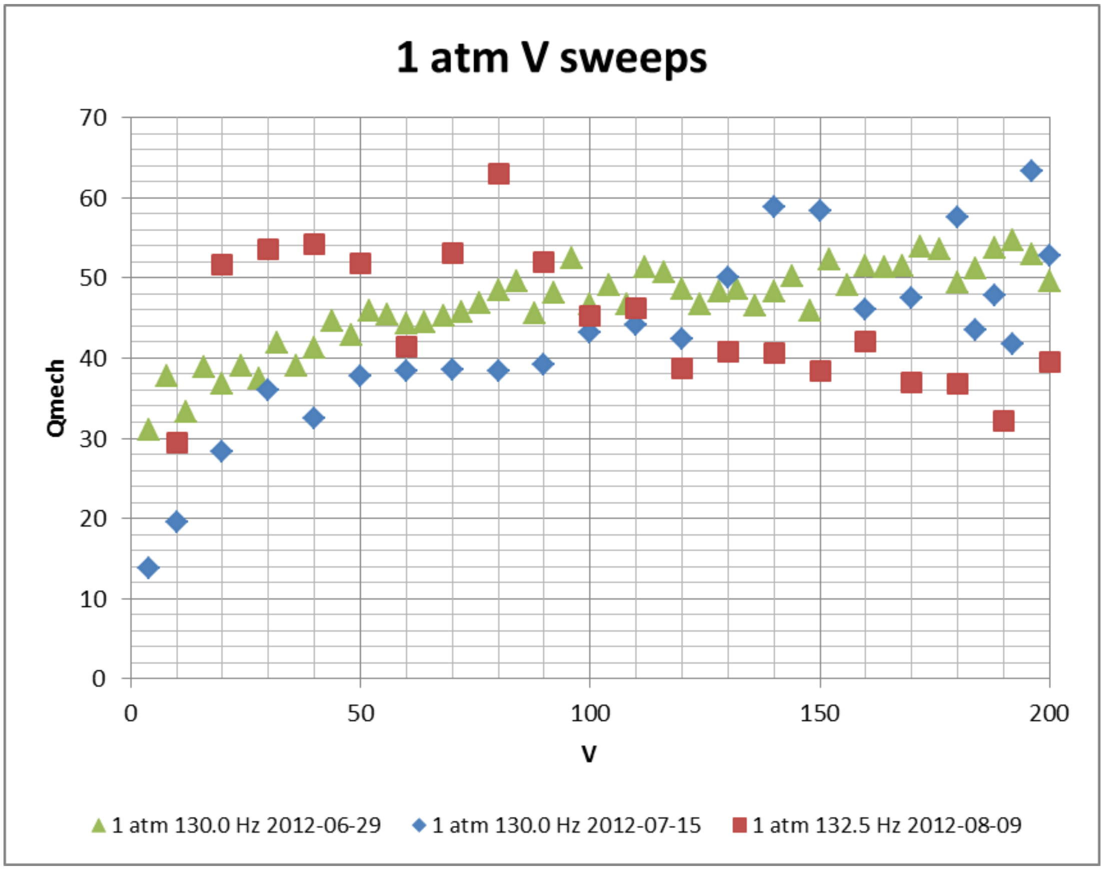


Appendix Figure O-4: Amplitude amplification vs. Voltage

Compared to the 15 Jul: 130 Hz V-sweep, the 9 August: 132.5 Hz V-sweep appears to give larger amplitudes at lower voltages and slightly lower amplitudes at higher voltages, consistent with the observation that lower voltages have a higher resonant frequency from the 5 Jul f-sweeps. However, the earlier 29 June: 130 Hz V-sweep did not reveal this, instead showing a smoothly increasing Ar0/Ar0DC as voltage decreased.

Several wing properties definitely changed over time - Most notably the default curvature. For the V-sweeps plotted here, this was:
2012-06-29: 117.04^o, stdev = 0.27^o.
2012-07-15: 106.88^o, stdev = 0.40^o.
2012-08-09: 104.96^o, stdev = 0.68^o.

If we use the flapping amplitude, frequency, and known wing size to determine the predicted drag coefficient using the approximate C_{DP} = 37.8/Re + 1.5, then use this to determine the predicted fluidic power dissipation, and finally use this predicted fluidic quality factor compared to the A0/A0DC to calculate a mechanical quality factor, we get the following:



Appendix Figure O-5: C_{DP} = 37.8/Re + 1.5

Section O.1.3.1 Making it better: use a specialty CDP fit for these V-sweeps

The C_{DP} = 37.8/Re + 1.5 is an OK fit to all data. However, by using a C_{DP} fit based only on data relevant around 130 Hz & 1 atm, we can compare to more specific data.

Equivalent frequencies for all the pressures:

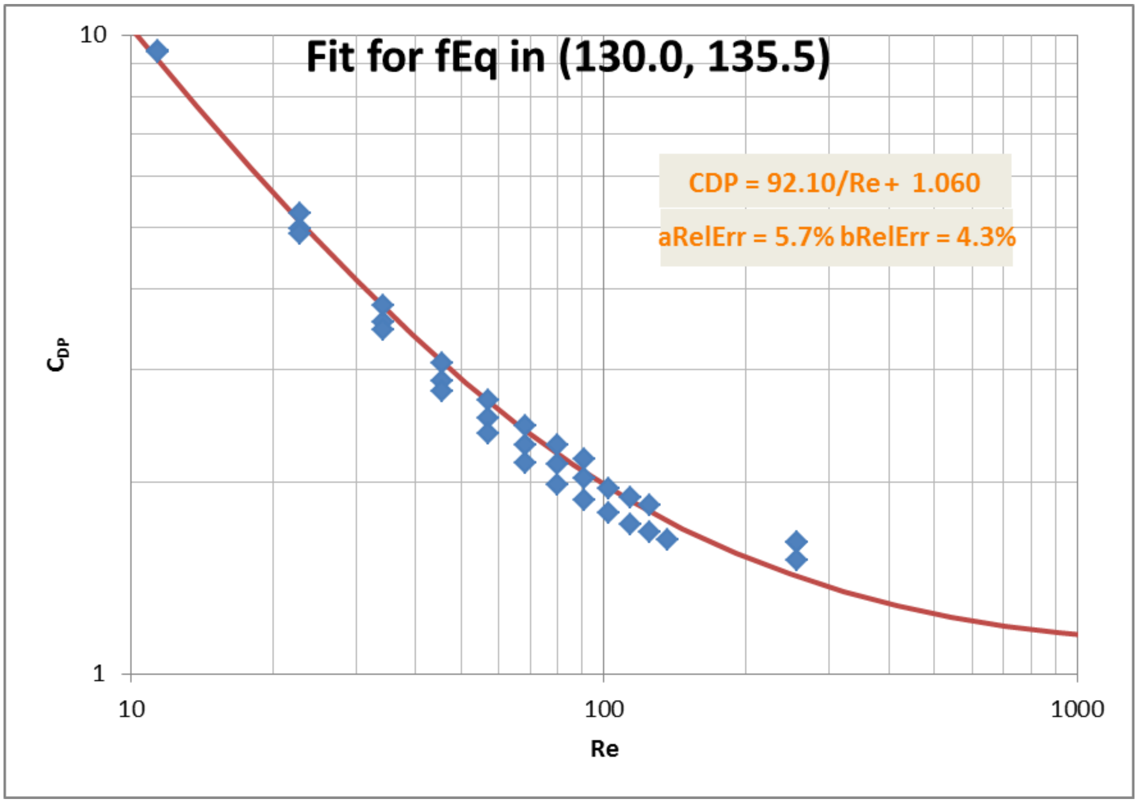
1 atm is 130-135.5

.7 atm: 91 - 95

.4 atm: 52.3 - 54.6

.2 atm: 26.2 - 27.5

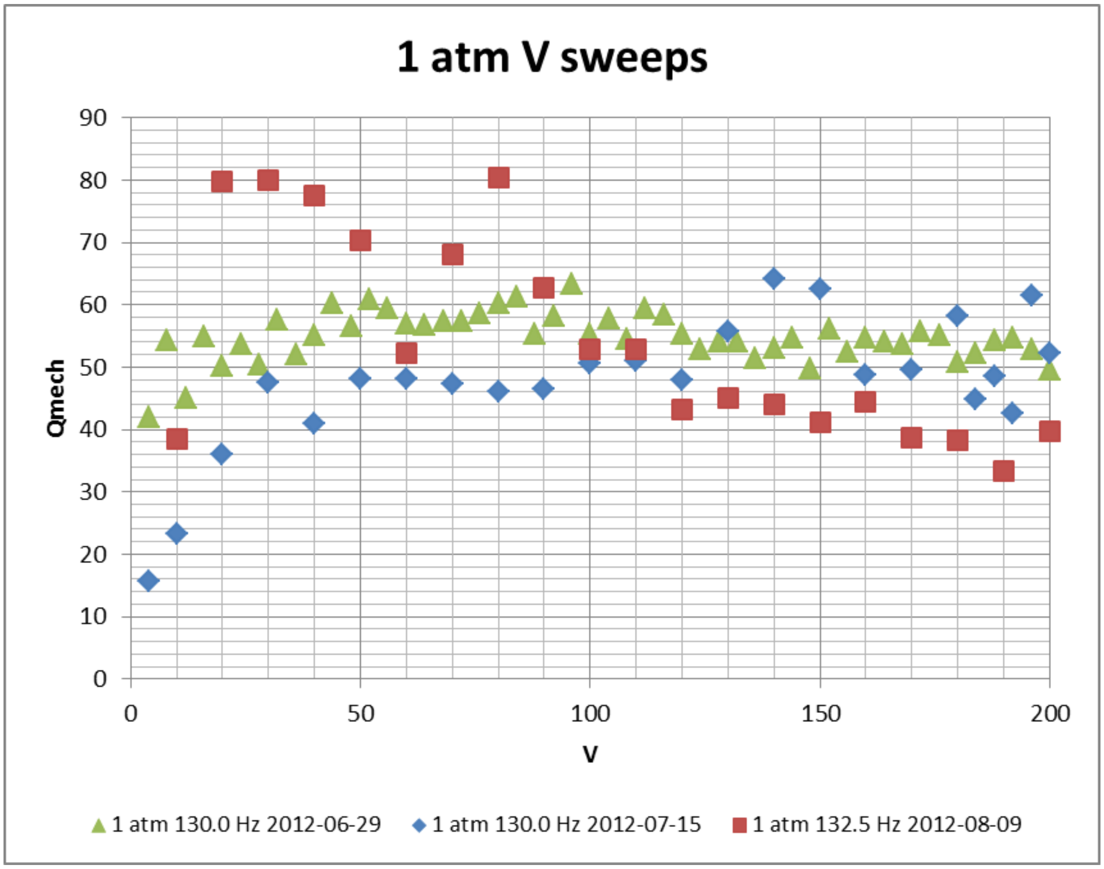
Using this data, the fit for C_{DP} vs Re actually tells a very different story:



Appendix Figure O-6: Drag coefficient for 1 atm sweeps

The trend line is $C_{DP} = \frac{92.1}{\text{Re}} + 1.06$. That's a LOT different than the global trend-line.

Using this fit instead for the drag coefficient now reveals a decreasing measured mechanical quality factor with voltage as well.



Appendix Figure O-7: Using C_{DP} = 92.1/Re + 1.06

On the one hand, this now suggests a decreasing mechanical quality factor with increasing amplitude, however it no longer decreases with *decreased* amplitude (save for the very low voltage at 130 Hz, where we know we're going off-resonance)

Section O.1.3.2 Making it better: correct amplitude for measuring off-resonance

If we use the resonance measures from 2012-07-05, we know that at 200 V, 130 Hz is the resonance but at 40 V, the resonance is higher; closer to 132 Hz. Use this information to come up with a voltage-specific "predicted A0 at resonance", and recalculate & plot the Q_{mech} using that.

Section 0.1.3.3 Making it better: separate wings

In reality, these amplitudes are the average of 2 wings. Plotting their amplitudes individually reveals that they resonate at close but not equal frequencies. By calculating the Q_{mech} for each wing individually, it may reveal new information, or force us to deal with extending the Q_{mech} theory to interference from partially overlapping resonances.

Section 0.2 0.7 atm:

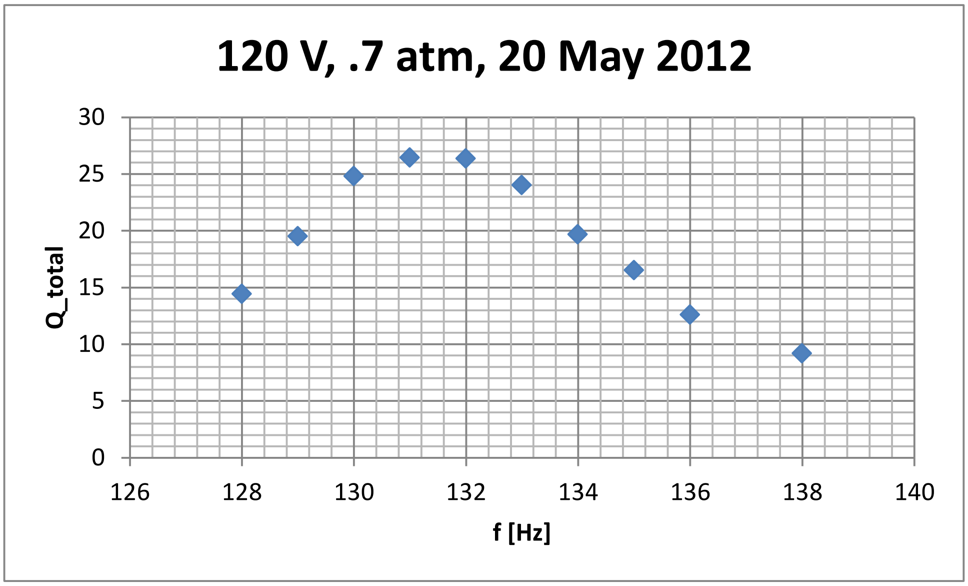
Section 0.2.1 20 May, 190-199, .7atm, f128-138

20 May

res @ 131

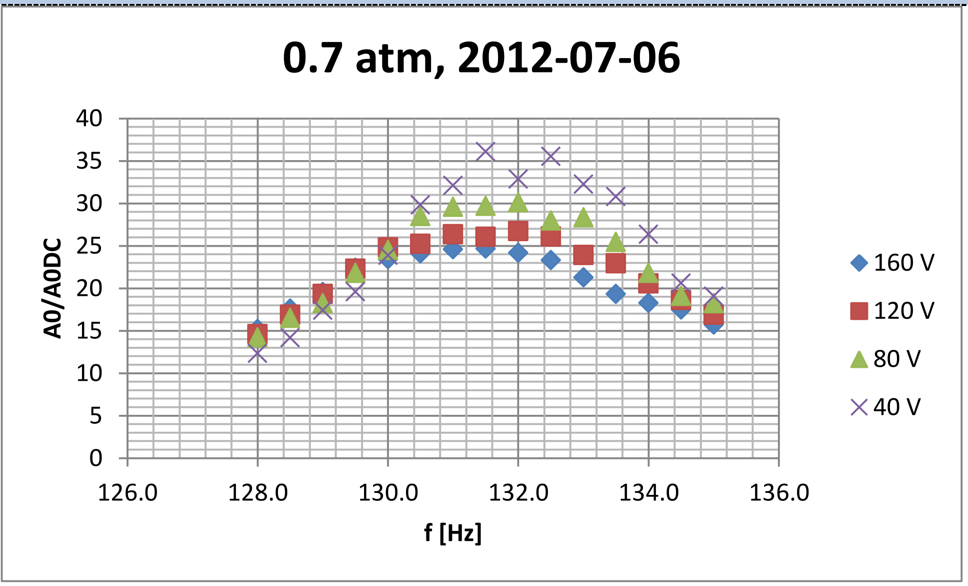
V=120, Ar0 = .1523

Qmech = 41.8



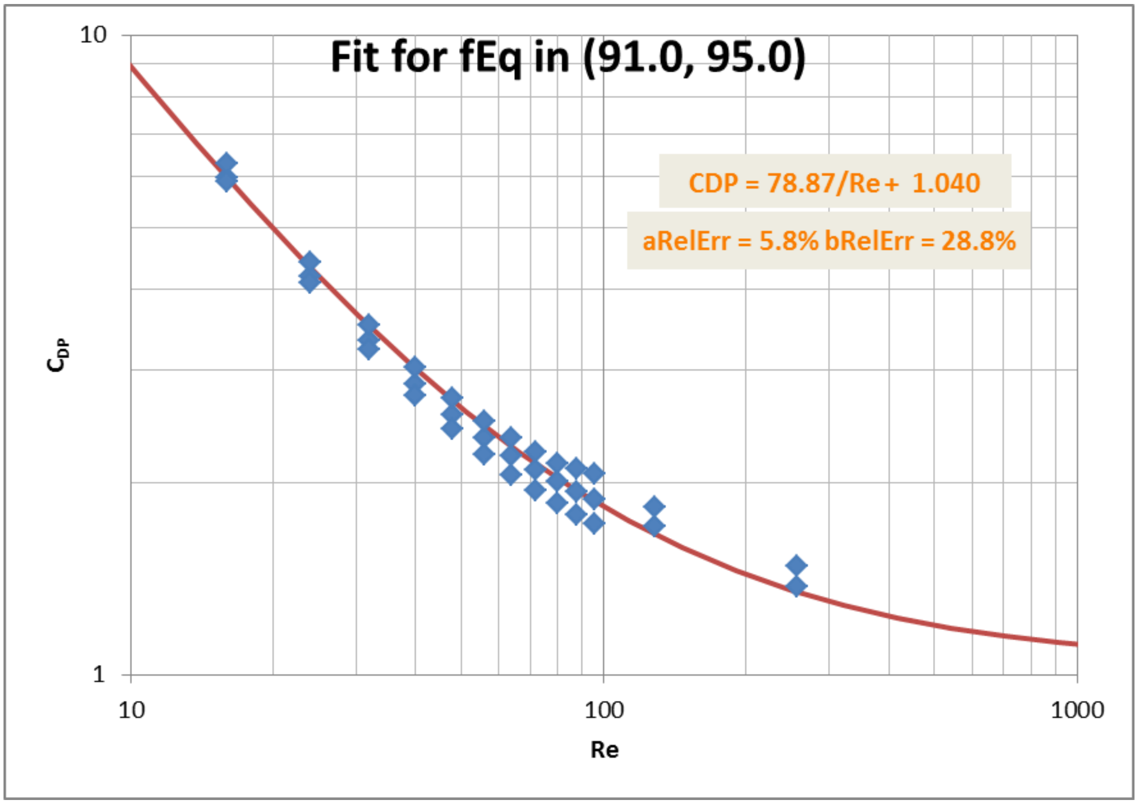
Appendix Figure O-8: Total quality factor for one sweep at 0.7 atm

Section 0.2.2 6 Jul

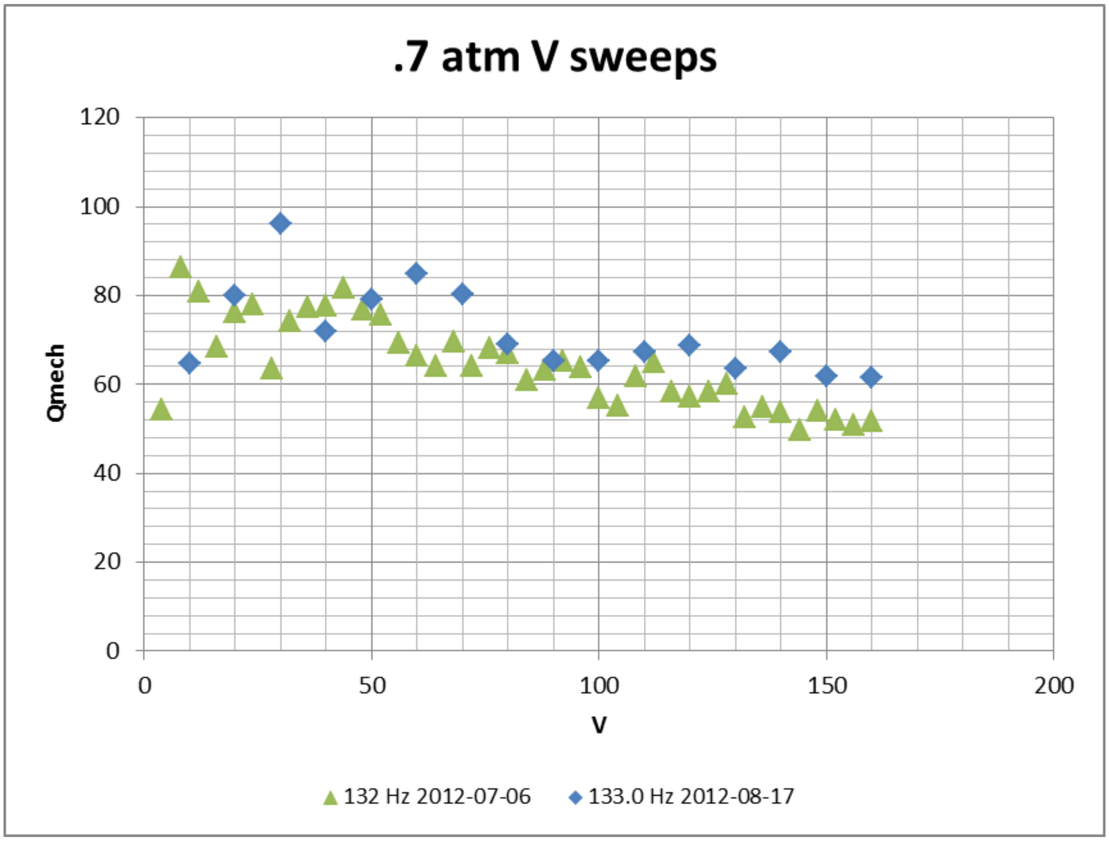


Appendix Figure O-9: Total quality factor from amplitude measures at 0.7 atm

.7atm:

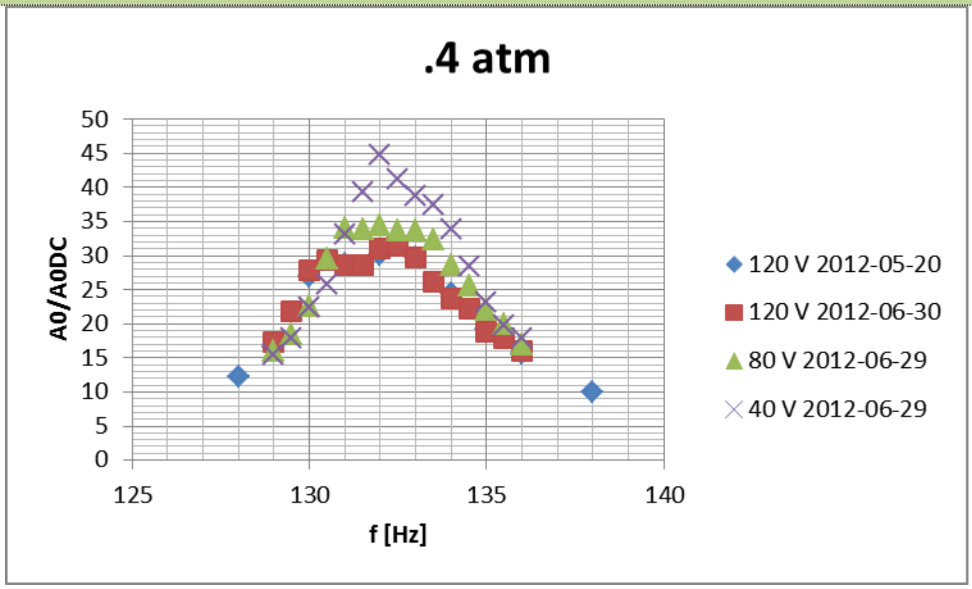


Appendix Figure O-10: Drag coefficient from CFX for 0.7 atm amplitude sweep range



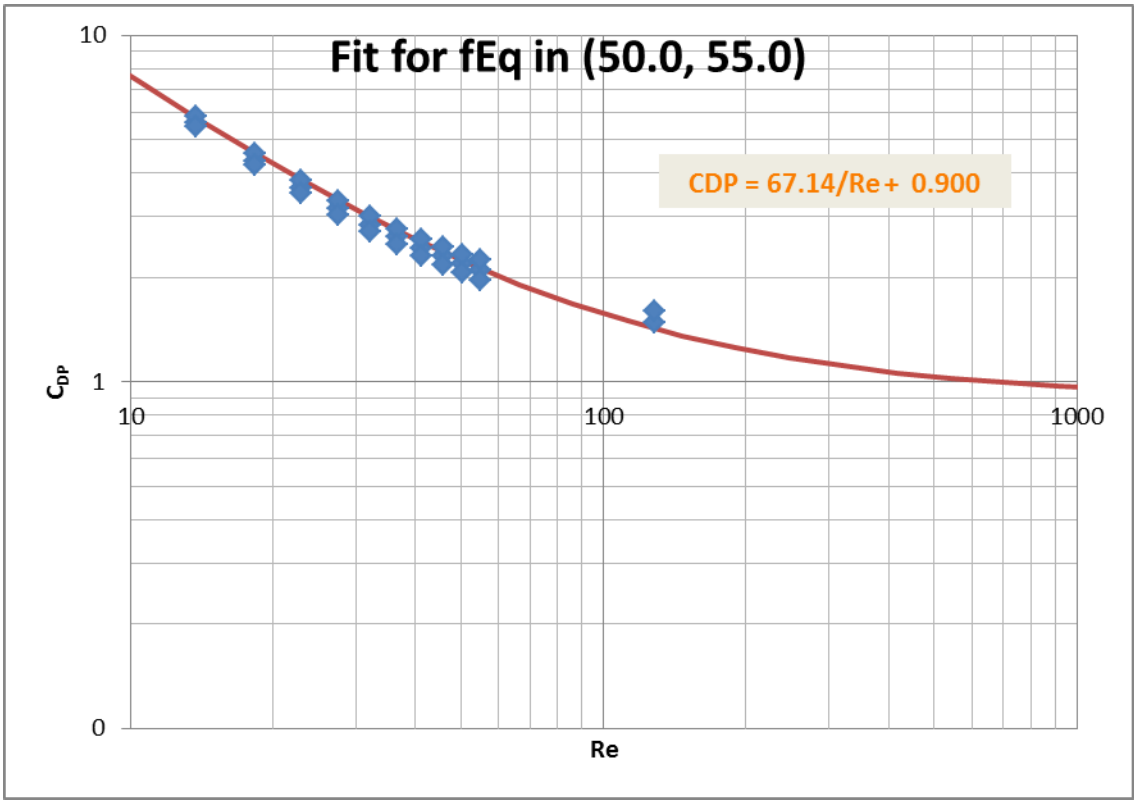
Appendix Figure O-11: Calculated mechanical quality factor for 0.7 atm

Section O.3 .4 atm:



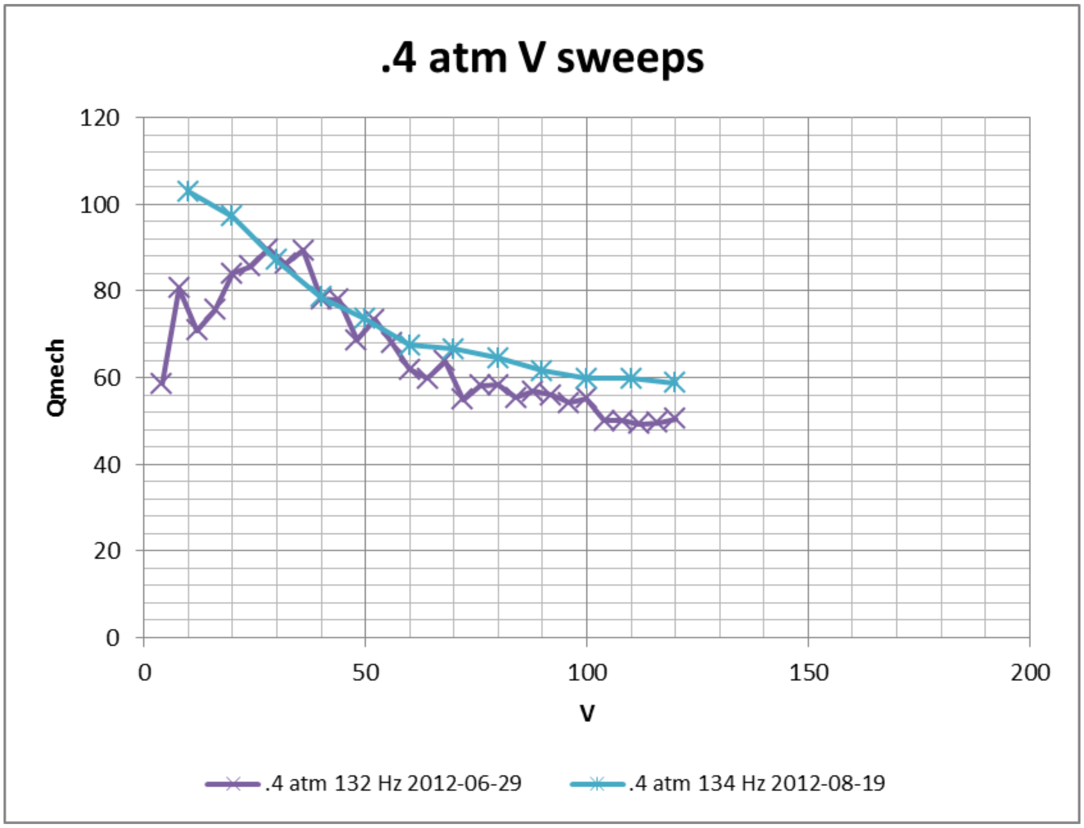
Appendix Figure O-12: Total quality factor from amplitude measures at 0.4 atm

Plotting the power drag coefficient for the fixed equivalent frequency applicable to the resonance measures in 0.4 atm, we obtain:



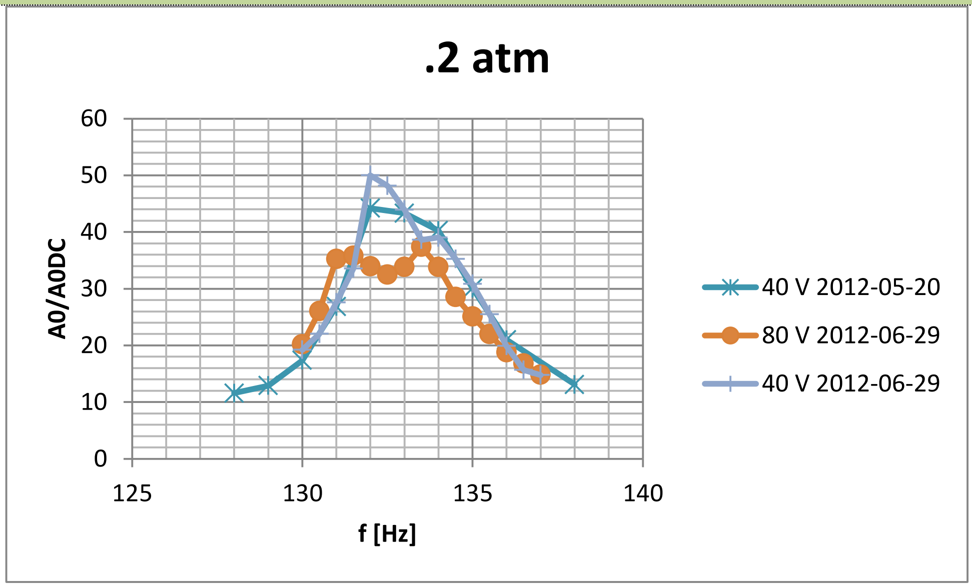
Appendix Figure O-13: Drag coefficient from CFX for 0.4 atm amplitude sweep range

Leading to these mechanical quality factors for the two relevant voltage sweeps at 0.4 atm:



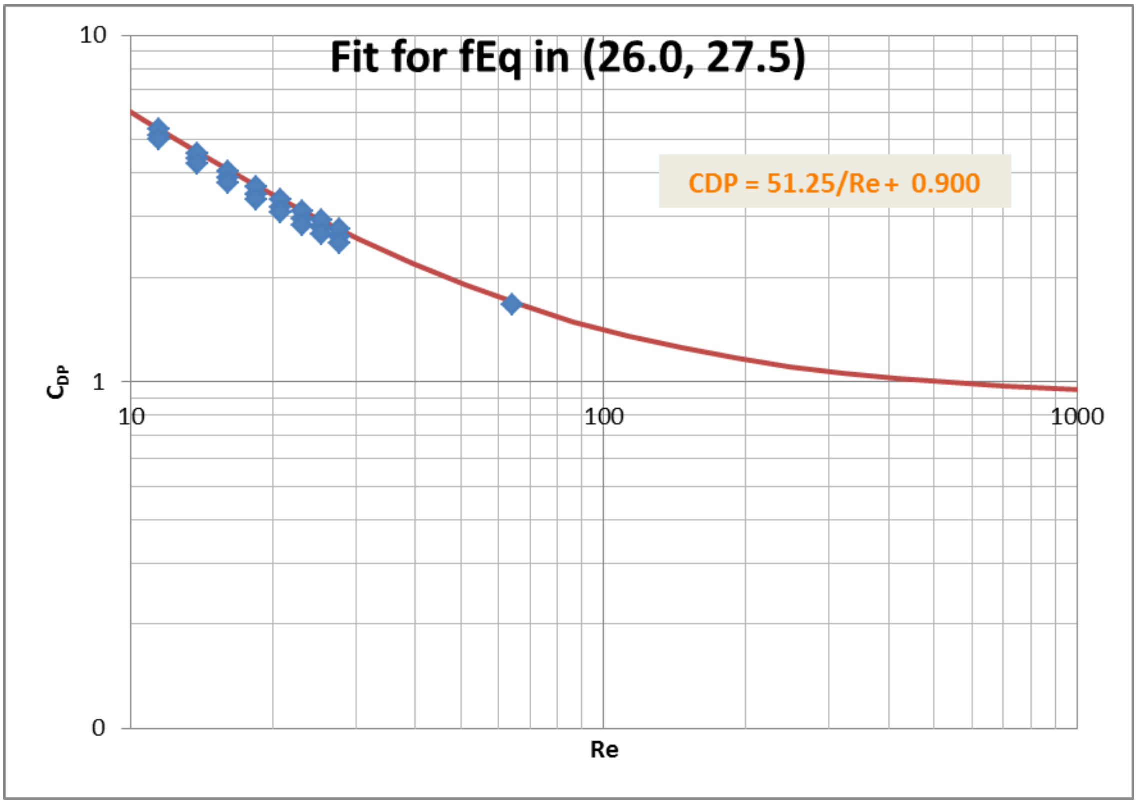
Appendix Figure O-14 Calculated mechanical quality factor for 0.4 atm

Section 0.4 .2 atm:



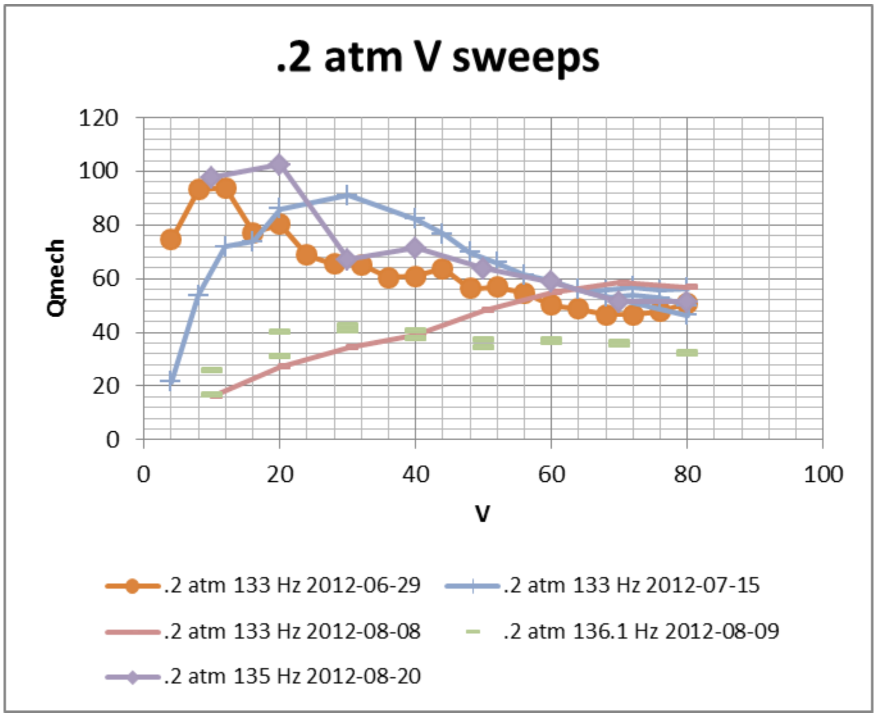
Appendix Figure O-15 Total quality factor from amplitude measures at 0.2 atm

As before, plotting the power drag coefficient for the fixed equivalent frequency applicable to the resonance measures in 0.2 atm, we obtain:



Appendix Figure O-16: Drag coefficient from CFX for 0.2 atm amplitude sweep range

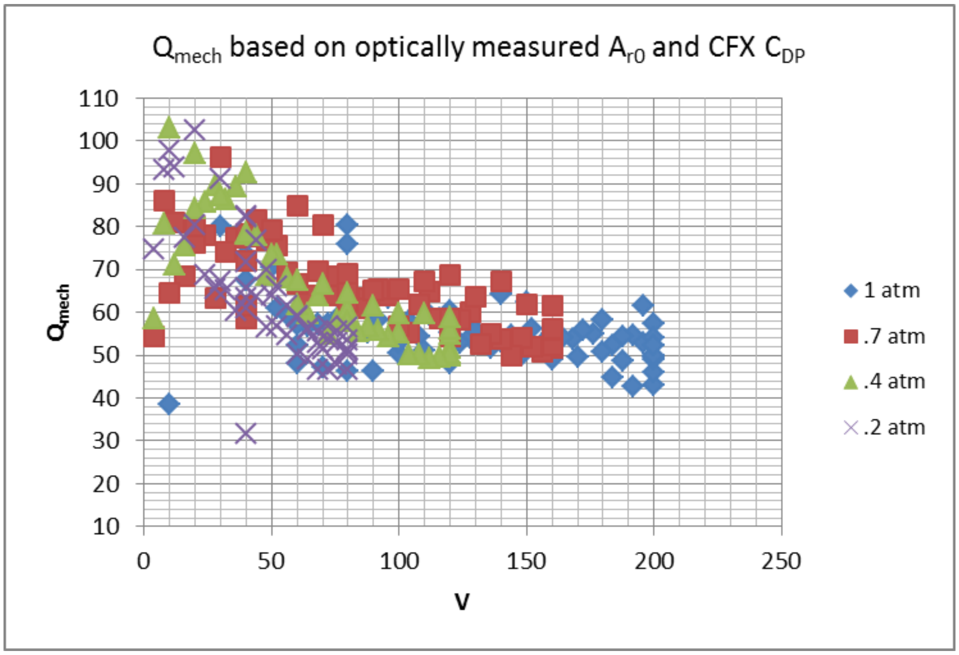
Which leads to these mechanical quality factors:



Appendix Figure O-17: Calculated mechanical quality factor for 0.2 atm

clearly, 136.1 Hz missed the resonance, and so did low-voltage 133 Hz at later dates.

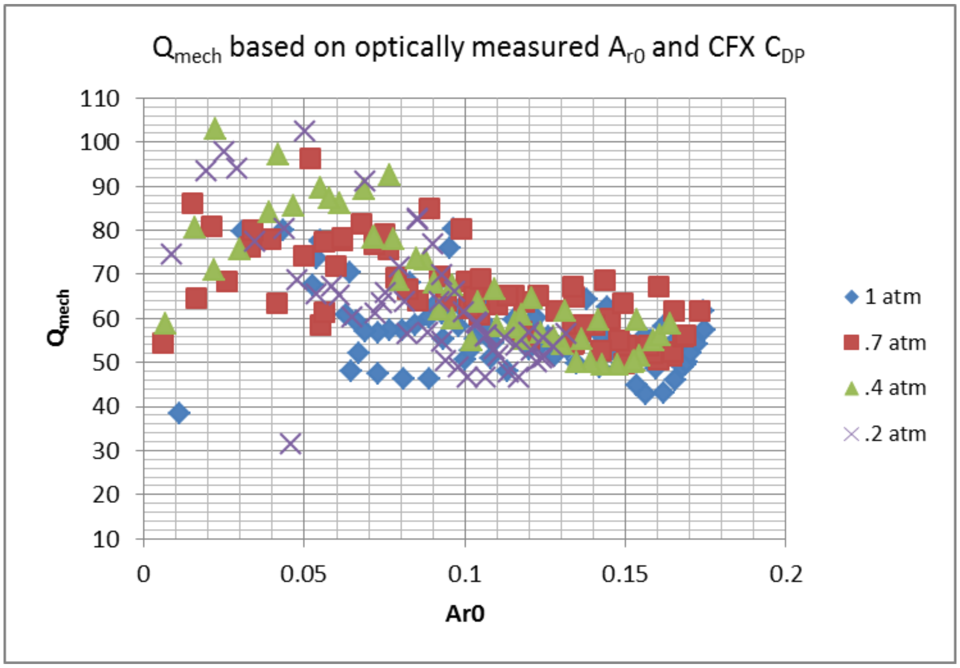
Section O.5 All pressures



Appendix Figure O-18: Mechanical quality factor vs. V for all pressures

(removed the off-resonance data)

This looks like there isn't too much difference between the different pressures. But to really compare them, we should plot vs. amplitude, not voltage. That plot looks like this:



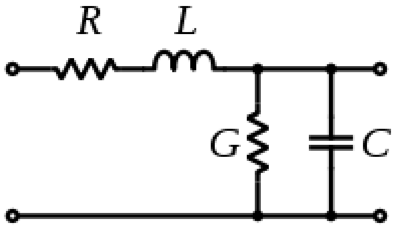
Appendix Figure O-19: Mechanical quality factor vs. A_{r0} for all pressures

No obvious difference appears between the different pressures: Mechanical quality factor is quite similar across different pressures and amplitudes, supporting the idea that the C_{DP} modeling is successful. Optically measuring very low amplitudes entailed a significant amount of uncertainty, so although the mechanical quality factor appears to rise as amplitude decreases, this may actually be due to a systematic error.

Appendix P Electrical Testing Procedure

Section P.1 Coax cable model

According to [79], a coaxial cable can be electrically modelled by



Appendix Figure P-1: Coaxial cable model from [79]

Where:

Shunt capacitance per unit length: $\frac{C}{h} = \frac{2\pi\varepsilon}{\ln\left(D/d\right)}$

Series inductance per unit length: $\frac{L}{h} = \frac{\mu}{2\pi}\ln\left(D/d\right) = \frac{\mu_0\mu_r}{2\pi}\ln\left(D/d\right)$

(h is length of cable, diameters are d (inner conductor) & D (shield)).

(series resistance is negligible most of the time)

Shunt conductance is very small for well-insulated cables.

All my stuff is at very low frequency compared to what these cables can handle. So it's doubtful they're responsible for anything strange.

Section P.2 Equivalent impedance of the BvD model:

In Chapter 4, we found the impedance of the BvD model at resonance. For testing purposes, we're going to need to know what this is off-resonance as well. So, let's derive the impedance of an RLC resonator in parallel with a package capacitance:

$$Z_{||} = Z_{r\equiv ResonantBranch} \,||\, Z_{p\equiv Package}$$

$$= \frac{Z_r Z_p}{Z_r + Z_p} = \frac{\left(R_1 + j\omega L_1 - \frac{j}{\omega C_1}\right)\frac{-j}{\omega C_p}}{R_1 + j\omega L_1 - \frac{j}{\omega C_1} - \frac{j}{\omega C_p}}$$

And at resonance, $\omega_1 L_1 = \dfrac{1}{\omega_1 C_1}$, therefore $L_1 = \dfrac{1}{\omega_1^2 C_1}$, and

$$\begin{aligned} Z_r &= R_1 + j\frac{\omega}{\omega_1^2 C_1} - \frac{j}{\omega C_1} \\ &= R_1 + \frac{j}{C_1}\left(\frac{\omega^2 - \omega_1^2}{\omega_1^2 \omega}\right) \\ &= R_1 + \frac{j}{\omega C_1}\left(\left(\frac{\omega}{\omega_1}\right)^2 - 1\right) \end{aligned}$$

making the parallel equivalent impedance:

$$\begin{aligned} Z_{\|} &= \frac{\left(R_1 + \dfrac{j}{\omega C_1}\left(\left(\dfrac{\omega}{\omega_1}\right)^2 - 1\right)\right)\dfrac{-j}{\omega C_p}}{R_1 + \dfrac{j}{\omega C_1}\left(\left(\dfrac{\omega}{\omega_1}\right)^2 - 1\right) - \dfrac{j}{\omega C_p}} \\ &= \frac{-j}{\omega C_p}\frac{\left(R_1 + \dfrac{j}{\omega C_1}\left(\left(\dfrac{\omega}{\omega_1}\right)^2 - 1\right)\right)}{R_1 + \dfrac{j}{\omega}\left(\dfrac{1}{C_1}\left(\left(\dfrac{\omega}{\omega_1}\right)^2 - 1\right) - \dfrac{1}{C_p}\right)} \end{aligned}$$

At low frequencies: $Z_{\|} \approx \dfrac{-j}{\omega C_p}\dfrac{\left(\dfrac{j}{\omega C_1}(-1)\right)}{\dfrac{j}{\omega}\left(\dfrac{1}{C_1}(-1) - \dfrac{1}{C_p}\right)} = Z_{C1} \| Z_p$

At high frequencies: $Z_{\|} \approx Z_p$

At resonance: $Z_{\|} = R_1 \| Z_p$

In terms of reactances:

$$X_r \equiv \frac{1}{\omega C_1}\left(\left(\frac{\omega}{\omega_1}\right)^2 - 1\right)$$

$$X_p \equiv \frac{-1}{\omega C_p}$$

Then:

$$Z_{\|} = jX_p \frac{R_1 + jX_r}{R_1 + j\left(X_r + X_p\right)}$$

Separate the real and imaginary parts by:

$$Z_{\|} = X_p \frac{jR_1 - X_r}{R_1 + j\left(X_r + X_p\right)} \times \frac{R_1 - j\left(X_r + X_p\right)}{R_1 - j\left(X_r + X_p\right)}$$

$$= X_p \frac{jR_1^2 + R_1\left(X_r + X_p\right) - X_r R_1 + jX_r\left(X_r + X_p\right)}{R_1^2 + \left(X_r + X_p\right)^2}$$

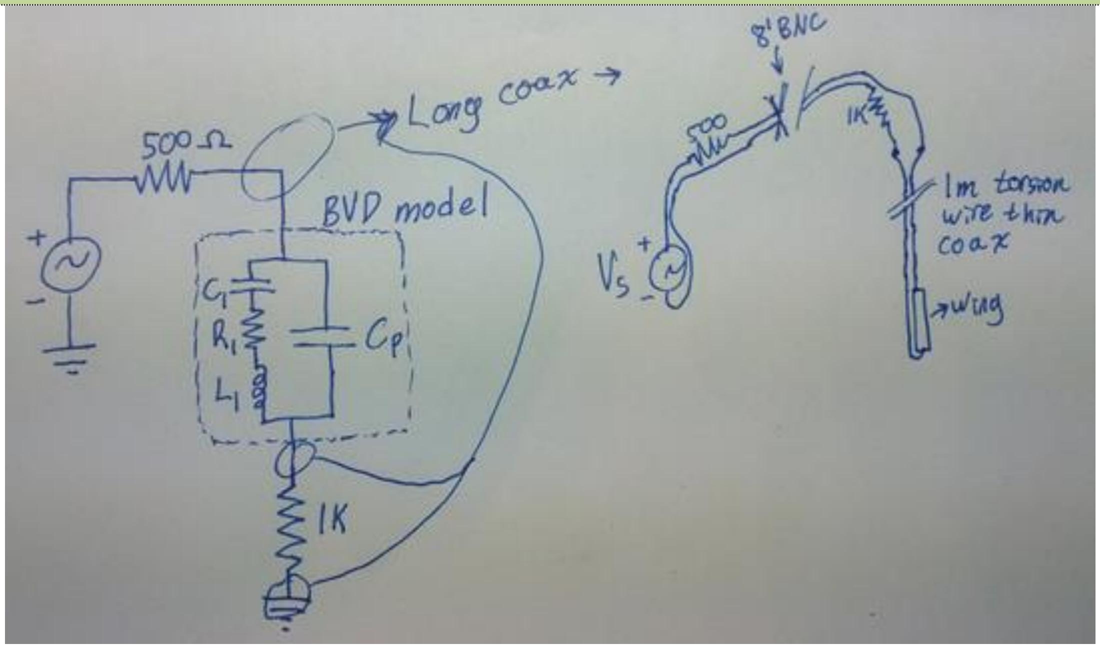
$$= X_p \frac{R_1 X_p + j\left(R_1^2 + X_r\left(X_r + X_p\right)\right)}{R_1^2 + \left(X_r + X_p\right)^2}$$

In conclusion:

$$\boxed{Z_{\|} = X_p \frac{R_1 X_p + j\left(R_1^2 + X_r\left(X_r + X_p\right)\right)}{R_1^2 + \left(X_r + X_p\right)^2}}$$

where: $X_p = \frac{-1}{\omega C_p}$ and $X_r = \frac{1}{\omega C_1}\left(\left(\frac{\omega}{\omega_1}\right)^2 - 1\right)$ are the reactances of the package and resonance branches, respectively.

Section P.3 Electrical Test Procedure: Current measures across a resistor out of the device



Appendix Figure P-2: Circuit model for measures

Let the 1k resistor be R$_s$ (the s is for sensing) Then:

$I = V_{Rs} / R_s$

In terms of the input voltage, $I = V_S / Z_{total}$ where $Z_{total} = R_s + Z_{\|} + R_L$ (where R_L is the "limiting" resistor)

V_s is known.

I is measured

I have theory on the parameters of Z$_{\|}$.

Referring to $\boxed{Z_{\|} = X_p \dfrac{R_1 X_p + j\left(R_1^2 + X_r\left(X_r + X_p\right)\right)}{R_1^2 + \left(X_r + X_p\right)^2}}$:

Using the notation:

1. $Z \equiv R + jX$: Impedance = Resistance + j*Reactance

2. $Y \equiv G + jB$: Admittance = Conductance + j*Susceptance

where $Y \equiv \frac{1}{Z}$ so that $G + jB = \frac{1}{R + jX} = \frac{R - jX}{R^2 + X^2}$

(Note that since conductance $G \equiv \mathrm{Re}(Y)$, the of a *branch* is **not** 1/ the resistance of that branch unless the branch has 0 reactance!)

Note that if R = 0, $jB = \frac{1}{jX} \to B = -\frac{1}{X}$)

Overall, current through the sense resistor is:

$$I = \frac{V_{Supply}}{Z_{total}} = \frac{V_{Supply}}{R_L + Z_{TS} + R_s}$$

where:

1. Test structure impedance is $Z_{TS} = \frac{1}{Y_{TS}}$
2. $Y_{TS} = Y_{res} + jB_{packageTS} + G_{LeakageTS}$
    a. $Y_{res} = \frac{1}{Z_{res}}$, and $Z_{res} = R_1 + jX_1$
        i. R_1 is the resonance branch resistance for the number of wings on the teststructure.
            1. Scales inversely with quality factor; proportional to damping.
        ii. $X_1 \equiv -\frac{1}{\omega C_1} + \omega L_1 = \frac{1}{\omega C_1} + \omega\left(\frac{1}{\omega_1^2 C_1}\right) = \frac{1}{\omega C_1}\left(\left(\frac{\omega}{\omega_1}\right)^2 - 1\right)$
            1. Independent of damping
    b. $B_{packageTS} = \frac{-1}{X_{packageTS}} = \omega C_{packageTS}$ is the package susceptance for the test structure from the BvD model.
        i. Output by WA4 / measured by capacitance bridge.

c. $G_{LeakageTS} = \dfrac{1}{R_{LeakageTS}}$ is the leakage conductance due to the piezoelectric layers not being perfect insulators.

i. Output by WA4 / measured by capacitance bridge.

Working in python, this is no problem: Python can do complex math and then plot the magnitude and phase at the end.

Input constants to get *I* at each frequency:

1. Teststructure:
   a. C1, R1, $C_{packageTS}$, $R_{LeakageTS}$, f_0,
2. Circuit:
   a. R_L, R_S.

Strategy: plot the *difference* between the measured current and predicted current for these constants and use it to fit the constants.

Section P.3.1 Detailed recipe using Difference measures:

Strategy:

1. Setup:
   a. Put capacitance bridge to match the wing circuit (except res branch) across the voltage output from the 1:20 amp.
   b. Measure current out of that by measuring voltage across a 1k: $V_A = I_{CB} R_{sCB}$
   c. Get very similar 1k resistor and use it to measure the current out of the wing: $V_B = I_{Wing} R_{sW}$
2. Each cycle:
   a. Set lockin to single input: A
   b. Autogain
   c. Measure lockin: V_A

        i. This measures current through the capacitance bridge:

        $I_{CB} = V_A / R_{sCB}$

    d. Calculate the impedance of the capacitance bridge: $Z_{CB} = R_{CB} - \frac{j}{\omega C_{CB}}$

    e. Calculate the 1:20 amp's output voltage: $V_S = V_A \times \frac{R_{sCB} + Z_{CB}}{R_{sCB}}$

    f. Set lockin to difference input: A-B

    g. Autogain

    h. Measure lockin: V_{A-B}

    i. Calculate $I_\Delta = \frac{-V_{A-B}}{R_{sCB} = R_{sW}}$ (output through the wing relative to just a capacitance bridge)

3. Output the phasors for:

    a. Z_{CB}

    b. V_s

    c. I_Δ

    d. $I_W = I_\Delta + \left(R_{sCB} + Z_{CB}\right) V_S$: instead of this, I output I_{CB}: $I_W = I_\Delta + I_{CB}$

4. Interpret the parameters:

    a. Likely Y_{CB} isn't exactly $Y_p + Y_{leakage}$. Supose the curve fits a resonance curve after additionally subtracting off the current through another admittance, Y_{fit}: $\begin{aligned} I_{\Delta fit} &= I_\Delta - I_{fit} \\ &= V_S\left(Y_W - Y_{CB} - Y_{fit}\right) \end{aligned}$

    b. therefore, $Y_p + Y_{leakage} = \left(Y_{CB} + Y_{fit}\right)$.

        i.

$$Y_p = \frac{1}{Z_p} = \frac{1}{1/\left(j\omega C\right)} = j\omega C$$

$$Y_{leakage} = \frac{1}{R_{leakage}}$$

ii. therefore $C_{package} = \text{Im}\left(\frac{Y_{CB}+Y_{fit}}{\omega}\right),\ R_{leakage} = \frac{1}{\text{Re}\left(Y_{CB}+Y_{fit}\right)}$

c. And for the resonance branch: $Z_{res} = R_1 + jX_1 = R_1 + \frac{j}{\omega C_1}\left(\left(\frac{\omega}{\omega_1}\right)^2 - 1\right)$

5. **Simplified final step:**

a. Let $Y_{Package+Leakage} = Y_{CB} + Y_{fit}$.

b. (ignoring the leakage, that's just: $j\omega C_{package} = j\omega C_{CB} + j\omega \Delta C_{package}$ -->

$\Delta C_{package} = C_{package} - C_{CB}$)

c. Since

i. $$\begin{aligned} &I_\Delta = I_{CB} - I_W \\ &\therefore I_W = I_{CB} - I_\Delta \end{aligned}$$

d. And $I_{res} = I_W - I_{package}$

i. $$\begin{aligned} &I_{res} = I_W - I_{package} \\ &I_{res} = \left(I_{CB} - I_\Delta\right) - \left(Y_{package}V_s\right) \\ &I_{res} = \left(Y_{CB} - Y_{package}\right)V_s - I_\Delta \\ &I_{res} = j\omega\left(C_{CB} - C_{package}\right)V_s - I_\Delta \\ &\therefore I_{res} = \left(-I_\Delta\right) - j\omega\left(C_{package} - C_{CB}\right)V_s \end{aligned}$$

ii. $\boxed{I_{res} = \left(-I_\Delta\right) - j\omega\left(\Delta C_{package}\right)V_s}$

e. e.g., set C$_{\text{package}}$ = 352, then $I_{res} = -I_\Delta - j\omega\left(352\text{pF} - 351\text{pF}\right)V_s$

f. Plot I_{res} vs. the *I* through a predicted Z_{res},

$$Z_{res} = R_1 + jX_1 = R_1 + \frac{j}{\omega C_1}\left(\left(\frac{\omega}{\omega_1}\right)^2 - 1\right)$$

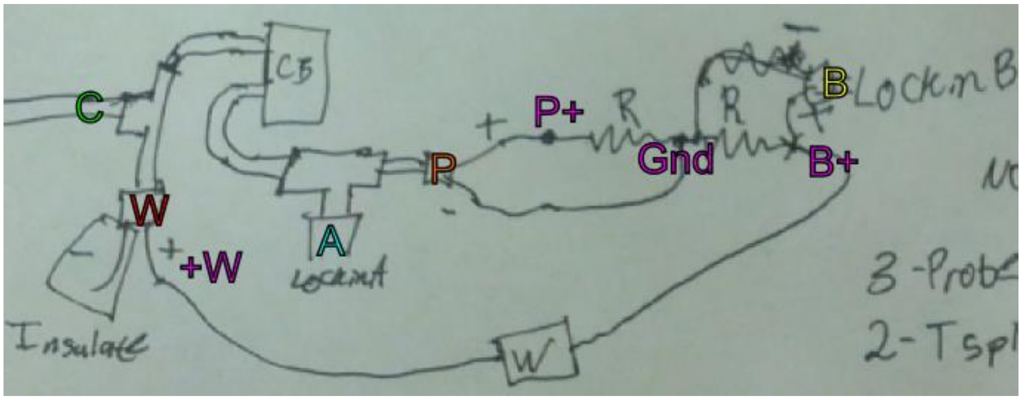
g. $I_{res} = \frac{V_s}{Z_{res}} = \frac{V_s}{R_1 + \frac{j}{\omega C_1}\left(\left(\frac{\omega}{\omega_1}\right)^2 - 1\right)}$

(imporant to use the fit V_s to not introduce new noise)

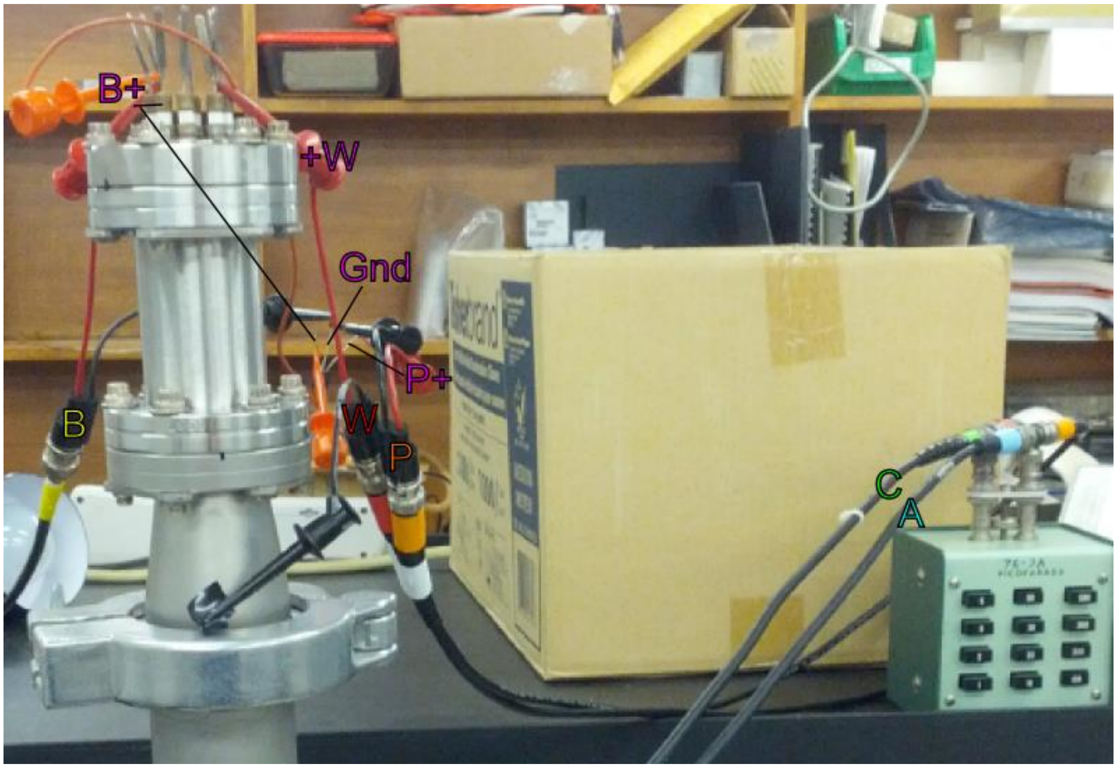
That's it! For each wing, compare the R1 & C1 that makes it fit the current difference curve nicely with their predictions.

Section P.3.1.1 Picture of the setup:

Colour-coded circuit map:



Appendix Figure P-3: Colour coded circuit model



Appendix Figure P-4: Colour coded realization of the circuit model

Section P.3.1.2 Decade capacitance box characterization

Set Cap [pF]	Meas Cap [pF]	Meas Loss [GOhm]
185	185.20059	1.41891
180	180.19173	1.42662
435	434.9359	0.6519
385	385.24622	0.71969

Section P.4 Electrical power losses

Summary

$P_R = \dfrac{V_{amp}^2}{2R}$, and $R = \dfrac{1}{2Qk_1}$ (at resonance), so that $P_R = Qk_1V_{amp}^2$.

Section P.5 Partial Vacuum Effect on Drag Force

The effect of partial vacuum on drag force is principally goverened by two dimensionless numbers:

1. Reynolds number $\mathrm{Re} \equiv \frac{\rho v L}{\mu}$, where ρ is the air density (proportional to pressure), v & L are the characteristic velocity and length scales, and μ is the [dynamic] viscosity (17.6e-6Pa-s for N_2 at 293K and independent of vacuum pressure).
    a. Re > 1: Form drag: $F_D = \frac{1}{2} C_D v^2 A$, where C_D is the drag coefficient and is only slowly varying or constant with Reynolds number, v is the velocity scale, and A is the frontal area.
    b. Re < 1: Stokes drag: $C_D = \frac{6\pi}{\mathrm{Re}}$ --> $F_D = 3\pi\mu L v$. Stokes drag force is *constant* with pressure as long as it's valid: Kn & Re < 1.
2. Knudsen number $\mathrm{Kn} \equiv \frac{\lambda}{L}$, where λ is the mean free path of gas molecules and L is the characteristic length scale.
    a. λ is inversely proportional to density; at room temperature & pressure, λ = 50nm.
    b. Kn << 1: Continuum regime
    c. Kn > 1: Free-Molecular Flow Regime: Drag force is proportional to pressure, velocity, and area: $F_{D\_FMR} = 0.01 P v A$ (the coefficient depends on the temperature and surface chemistry). This equals the stokes drag force for an L = 2mm object at Kn = 0.3.

The Knudsen number is proportional to the ratio of Mach and Reynolds numbers: $\mathrm{Kn} = \frac{\mathrm{Ma}}{\mathrm{Re}}\sqrt{\frac{\gamma\pi}{2}}$, where γ is the ratio of the gas's specific heat at constant pressure to that at constant volume, [80].

Appendix Q MATLAB Program to process the Zygo Data

```
%Made on 29 Feb 2012 to load and process datafiles output by Metropro when
%using the optical profilometer
%Generated using MatlabPrograms.doc in Dropbox\General Reference\Programming\
clear all;
close all;
disp('****ZygoProcStart*********************************')

%ASCII file to read:
DatFileName='Su8_Gold_2.asc';
Data3dPlot=0;
%Tilt correction points
FitLowerLeft=[180,465];
FitUpperRight=[200,475];
NumberOfPointsPerFitLine = 5;

%Points To Calculate Height At:
TiltCorrectedPlot=1;
TiltCorrectedContourNumber=100;
xHeightPoints=[200,205,170,170,288,292,297,100, 100, 100, 600, 600];
yHeightPoints=[475,475,261, 262, 107,102,95,250, 50, 400, 50, 400];

xHeightPoints2=xHeightPoints+1;
yHeightPoints2=yHeightPoints;
xHeightPoints3=xHeightPoints;
yHeightPoints3=yHeightPoints+1;
EtchCalcs = 0;  %Turn off etch rate calculations if this is not an etch depth test
RawDataPlot=0;
RawDataContourNumber=20;

PointsToAverage = 4:7;


DatFID=fopen(DatFileName, 'r');
if(DatFID == -1)
    fprintf('Missing DatFile: %s.\n', DatFileName);
    break;
end

%% Read the headers
%Expected syntax of an output ASCII metropro file is described in MetroPro Reference Guide
J.pdf:
%1 Zygo ASCII Data File - Format 2
Header1=fgetl(DatFID);
%2 SoftwareType  MajorVers  MinorVers  BugVers  SoftwareDate
Header2=fgetl(DatFID);
%3 IntensOriginX  IntensOriginY  IntensWidth  IntensHeight  NBuckets  IntensRange
%e.g., 0 0 640 480 1 255
Header3=fscanf(DatFID, '%d %d %d %d %d %d', 6);
IntensWidth=Header3(3);
IntensHeight=Header3(4);

%4 PhaseOriginX  PhaseOriginY  PhaseWidth  PhaseHeight
%e.g., 0 0 640 480
Header4=fscanf(DatFID, '%d %d %d %d ', 4);
PhaseWidth=Header4(3);
PhaseHeight=Header4(4);

%5 Comment
fgetl(DatFID);
%6 PartSerNum
fgetl(DatFID);
```

```
%7 PartNum
fgetl(DatFID);
%8 Source  IntfScaleFactor  WavelengthIn  NumericAperture  ObliquityFactor Magnification
CameraRes  TimeStamp
%e.g., 0 0.5 6.48e-007 0 1 0 2.85262e-006 1329236048
Header8=fscanf(DatFID, '%d %g %g %d %d %g %g %d', 8);

%9 CameraWidth  CameraHeight  SystemType  SystemBoard  SystemSerial InstrumentId
ObjectiveName
%e.g., 640 480 11 0 50299 0 " 10X  Mirau"
Header9=fscanf(DatFID, '%d %d %d %d %d %d ', 6);
ObjectiveName=fgetl(DatFID);
%10 AcquireMode  IntensAvgs  PZTCal  PZTGain  PZTGainTolerance  AGC TargetRange
LightLevel  MinMod  MinModPts
%e.g., 2 0 0 0 0 0 0 30.7937 1 0
Header10=fscanf(DatFID, '%d %d %d %d %d %d %d %g %d %d ', 10);
%11 PhaseRes  PhaseAvgs  MinimumAreaSize  DisconAction  DisconFilter ConnectionOrder
RemoveTiltBias  DataSign  CodeVType
%e.g., 2 2 7 0 0 0 0 0 0
Header11=fscanf(DatFID, '%d %d %d %d %d %d %d %d %d ', 9);
%12 SubtractSysErr  SysErrFile
%e.g., 0 "                            "
Header12=fscanf(DatFID, '%d ', 1);
SysErrFile=fgetl(DatFID);
%13 RefractiveIndex  PartThickness
%e.g., 1 0
Header13=fscanf(DatFID, '%d %d ', 2);
%14 ZoomDesc
Header14=fgetl(DatFID);
fgetl(DatFID); %clear the '#' sign.

IntensityScaleFactor=Header8(2);
ObliquityFactor=Header8(5);
CameraRes=Header8(7);  %Tells you m/pixel
lambda=Header8(3);
PhaseRes=Header11(1);
if(PhaseRes==0)
    ResolutionCorrection=4096;
elseif(PhaseRes==1)
    ResolutionCorrection=32768;
elseif(PhaseRes==2)
    ResolutionCorrection=131072;
else
    disp(['Invalid PhaseRes in datafile: ', num2str(PhaseRes)])
    break
end

fprintf('Intensity data %dx%d, Phase data %dx%d\n', IntensWidth, IntensHeight, PhaseWidth,
PhaseHeight);
fprintf('S = %g, O = %d, lambda = %gnm\n', IntensityScaleFactor, ObliquityFactor,
lambda*1e9);
fprintf('ResolutionCorrection = %d\n', ResolutionCorrection);

%% Read the bulk of the datafile
IntensData=fscanf(DatFID, '%g ', [IntensWidth, IntensHeight]);
fgetl(DatFID) %display a '#'
PhaseData=fscanf(DatFID, '%g ', [PhaseWidth, PhaseHeight]);
fgetl(DatFID) %display a '#'
PhaseDataProcessed=PhaseData.*(lambda*IntensityScaleFactor*ObliquityFactor/ResolutionCorre
ction);

%% Plot the heights
%Preallocation for speed
% x=zeros(1,PhaseWidth*PhaseHeight);%init
% y=zeros(1,PhaseWidth*PhaseHeight);%init
% z=zeros(1,PhaseWidth*PhaseHeight);%init
% %2d init
% z2=zeros(PhaseWidth,PhaseHeight);%init
% %Generate datapoints
% for(rowCounter=1:PhaseHeight)
%     for(colCounter=1:PhaseWidth)
%         y((rowCounter-1)*PhaseWidth+colCounter)=PhaseHeight+1-rowCounter;
%         x((rowCounter-1)*PhaseWidth+colCounter)=colCounter;
%         if(PhaseData(colCounter,rowCounter) <2147483640) %If this datapoint was a valid
reading
%             z((rowCounter-
1)*PhaseWidth+colCounter)=PhaseDataProcessed(colCounter,rowCounter); %as a 1d array
%         else
%             z((rowCounter-1)*PhaseWidth+colCounter)=0; %as a 1d array
%         end
%     end
% end
%2d version:
z=zeros(PhaseWidth,PhaseHeight);%init
for(rowCounter=1:PhaseHeight)
    for(colCounter=1:PhaseWidth)
        if(PhaseData(colCounter,rowCounter) <2147483640) %If this datapoint was a valid
reading
            z(colCounter,PhaseHeight+1-
rowCounter)=PhaseDataProcessed(colCounter,rowCounter);  %as a 2d array
        else
            z(colCounter,PhaseHeight+1-rowCounter)=0;  %as a 2d array
        end
    end
end


nx=size(z,1); %number of x-points
ny=size(z,2); %number of y-points
if(Data3dPlot)
    xvec=zeros(1,nx*ny);
    yvec=zeros(1,nx*ny);
    zvec=zeros(1,nx*ny);
    for(cx=1:nx)
        for(cy=1:ny)
            xvec((cx-1)*ny+cy)=cx;
            yvec((cx-1)*ny+cy)=cy;
            zvec((cx-1)*ny+cy)=z(cx,cy);
        end
    end

    plot3(xvec,yvec,zvec, '.')
    xlabel('x')
    ylabel('y')
    zlabel('z')
end
fclose(DatFID);

if(RawDataPlot)
    figure(2);
    contourf(z',RawDataContourNumber);
    xlabel('x')
    ylabel('y')
    PltTitleStr = strrep(DatFileName, '_', '\_'); %Makes underscores in the file name show
up correctly in the plot title.
    title(['Raw Height Data: ', PltTitleStr]);
    grid on;
    hold on;
    plot(xFitPoints,yFitPoints,'g*')
end
```

```
%% Calculate the tilt correction
%First, find the coordinates of the points to use on each fit line:
xFitPoints=round(linspace(FitLowerLeft(1), FitUpperRight(1), NumberOfPointsPerFitLine));
yFitPoints=round(linspace(FitLowerLeft(2), FitUpperRight(2), NumberOfPointsPerFitLine));

[X,Y] = meshgrid(xFitPoints, yFitPoints)
for row = 1:NumberOfPointsPerFitLine
    for col = 1:NumberOfPointsPerFitLine
        Z(row,col) = z(xFitPoints(col),yFitPoints(row))
    end
end



A = ones(6);
d = ones(6,1);
xCoefPows = [2,1,0,1,0,0];
yCoefPows = [0,1,2,0,1,0];
for row = 1:6
    for col = 1:6
        A(row,col) =
sum(sum(X.^(xCoefPows(row)+xCoefPows(col)).*Y.^(yCoefPows(row)+yCoefPows(col))));
    end
    d(row) = sum(sum(Z.*X.^(xCoefPows(row)).*Y.^(yCoefPows(row))));
end
%Fit Coefficients
p = A\d

%Then the fit parabolas are z = a(1)+a(2)*x+a(3)*y+a(4)*x^2+a(5)*y^2+a(6)*x*y, so correct
the
%z-points like this:
zCor=z;%init
for(cx=1:nx)
    for(cy=1:ny)
        if(z(cx,cy)>0)
            zCor(cx,cy)=z(cx,cy)-(p(1)*cx^2+p(2)*cx*cy+p(3)*cy^2+p(4)*cx+p(5)*cy+p(6));
%Tilt corrected z
        end %else, leave it as 0
    end
end





%plot3(x,y,z, 'r.')
%plot3(xFitPoints,yFitPoints,zFitPoints,'r*')

%% Plotting and outputs
if(TiltCorrectedPlot)
    figure(3);
    contourf(zCor', TiltCorrectedContourNumber);
    xlabel('x')
    ylabel('y')
    PltTitleStr = strrep(DatFileName, '_', '\_'); %Makes underscores in the file name show
up correctly in the plot title.
    title(['Curvature-Corrected Height Data: ', PltTitleStr]);
    grid on;

    PointLabels={'R'};

    %RefHeight=zCor(xHeightPoints(1),yHeightPoints(1)); %Height reference is the first
point
    RefHeight1=sum(sum(zCor((xHeightPoints(1)-1):(xHeightPoints(1)+1),(yHeightPoints(1)-
1):(yHeightPoints(1)+1))))/9; %Height reference is the first point
```

```
    RefHeight2=sum(sum(zCor((xHeightPoints2(1)-
1):(xHeightPoints2(1)+1),(yHeightPoints2(1)-1):(yHeightPoints2(1)+1)))))/9; %Height
reference is the first point
    RefHeight3=sum(sum(zCor((xHeightPoints3(1)-
1):(xHeightPoints3(1)+1),(yHeightPoints3(1)-1):(yHeightPoints3(1)+1)))))/9; %Height
reference is the first point
    RefHeight=(RefHeight1+RefHeight2+RefHeight3)/3;
    fprintf('Abs height at the reference points = (%g,%g,%g)m, average = %g m\n',
RefHeight1, RefHeight2,RefHeight3, RefHeight);
    fprintf('Rel height at the reference points = (%g,%g,%g)um, average = %g m\n',
(RefHeight1-RefHeight)*1e6, (RefHeight2-RefHeight)*1e6,(RefHeight3-RefHeight)*1e6, 0);


    fprintf('\nRelative depths of each point, along with the average & stdev of all 3
colours for that point:\n')
    fprintf('%3s%9s%9s%9s%9s%9s\n', 'P#', 'D1', 'D2', 'D3', 'Avg', 'StDev')
    fprintf('%3s%9s%9s%9s%9s%9s\n', '#', 'um', 'um', 'um', 'um', 'um')
    for(c=2:length(xHeightPoints))
        PointLabels{c}=num2str(c);
        %RelHeight(c)=zCor(xHeightPoints(c),yHeightPoints(c))-RefHeight;
        RelHeight1(c)=sum(sum(zCor((xHeightPoints(c)-
1):(xHeightPoints(c)+1),(yHeightPoints(c)-1):(yHeightPoints(c)+1)))))/9-RefHeight; %use 9pt
avg
        RelHeight2(c)=sum(sum(zCor((xHeightPoints2(c)-
1):(xHeightPoints2(c)+1),(yHeightPoints2(c)-1):(yHeightPoints2(c)+1)))))/9-RefHeight; %use
9pt avg
        RelHeight3(c)=sum(sum(zCor((xHeightPoints3(c)-
1):(xHeightPoints3(c)+1),(yHeightPoints3(c)-1):(yHeightPoints3(c)+1)))))/9-RefHeight; %use
9pt avg
        RelHeight(c)=(RelHeight1(c)+RelHeight2(c)+RelHeight3(c))/3;
        stdev(c) = sqrt(((RelHeight1(c)-RelHeight(c))^2+(RelHeight2(c)-
RelHeight(c))^2+(RelHeight3(c)-RelHeight(c))^2)/2);
        %fprintf('#%d pts rel depth = (%g,%g,%g)um, Avg=%g um, stdev = %.1e
um\n',c,RelHeight1(c)*1e6, RelHeight2(c)*1e6,RelHeight3(c)*1e6, RelHeight(c)*1e6,
stdev(c)*1e6);
        fprintf('%3d%9.4f%9.4f%9.4f%9.4f%9.4f\n',c,RelHeight1(c)*1e6,
RelHeight2(c)*1e6,RelHeight3(c)*1e6, RelHeight(c)*1e6, stdev(c)*1e6);
    end

    fprintf('\nAveraging these points:')
    disp(PointsToAverage)
    fprintf('find\nmean = %g um, stdev = %g um\n', mean(RelHeight(PointsToAverage))*1e6,
std(RelHeight(PointsToAverage))*1e6)

    hold on;
    text(xHeightPoints,yHeightPoints,PointLabels, 'BackgroundColor', 'w');
    text(xHeightPoints2,yHeightPoints2,PointLabels, 'BackgroundColor', 'm');
    text(xHeightPoints3,yHeightPoints3,PointLabels, 'BackgroundColor', 'c');
    %Plot the points used for the curvature correction:
    plot(X,Y, 'g*')


    if(EtchCalcs)
        AvgEtchRate=min(RelHeight)/sum(EtchTimes)
        n=length(EtchTimes);
        EtchComboMatrix=zeros(2^n-1,n);
        EtchNames=[];
        for(crow=1:(2^n-1))
            EtchNames{crow}=[];
            for(ccol=1:n)
                EtchComboMatrix(crow,ccol)=mod(floor(crow/2^(n-ccol)),2);
                if(EtchComboMatrix(crow,ccol)>0)
                    if(length(EtchNames{crow})==0)  %if first entry for this one
                        EtchNames{crow}=num2str(ccol);
                    else
```

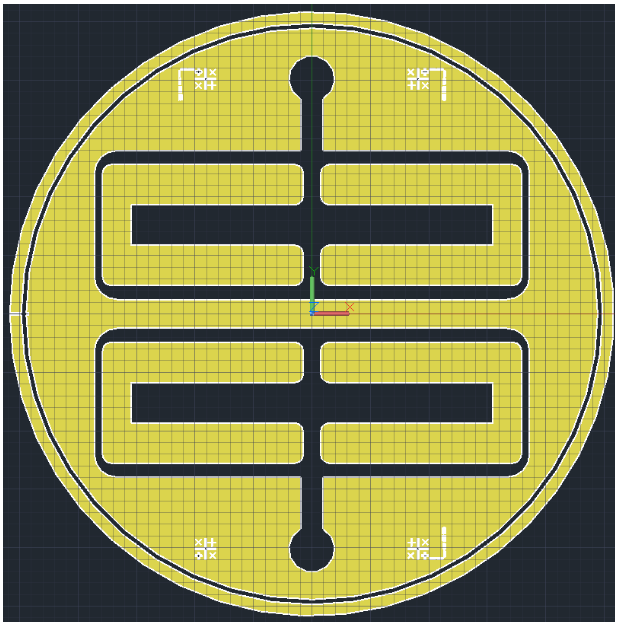
```
                        EtchNames{crow}=[EtchNames{crow}, ' and ', num2str(ccol)];
                    end
                end
            end
        end
        PossibleEtchTimes=EtchComboMatrix*EtchTimes';
        %EtchNames

        for(c=2:length(RelHeight))
            TimeEstimate(c)=RelHeight(c)/AvgEtchRate;
            BestEtchComboGuessNumber=1;
            BestCloseness=abs(PossibleEtchTimes(1)-TimeEstimate(c))/TimeEstimate(c);
            for(cEtchTimes=2:length(PossibleEtchTimes))
                Closeness=abs(PossibleEtchTimes(cEtchTimes)-
TimeEstimate(c))/TimeEstimate(c);
                if(Closeness<BestCloseness)
                    BestCloseness=Closeness;
                    BestEtchComboGuessNumber=cEtchTimes;
                end
            end
            BestEtchComboTime(c)=PossibleEtchTimes(BestEtchComboGuessNumber);
            BestEtchComboName{c}=EtchNames{BestEtchComboGuessNumber};
            fprintf('Pt #%d @(%d,%d) = %g um, TimeGuess=%g hr,ClosestTime=%g hr from
Etches %s\n',c,xHeightPoints(c),yHeightPoints(c), RelHeight(c)*1e6, TimeEstimate(c),
BestEtchComboTime(c), BestEtchComboName{c});
        end
        for(c=2:length(EtchNumberManualSpec))
            ManualEtchTime(c)=PossibleEtchTimes(EtchNumberManualSpec(c));
            ManualEtchNames{c}=EtchNames{EtchNumberManualSpec(c)};
        end

        LayerEtchRate=(RelHeight(2:end)*1e6)./ManualEtchTime(2:end);
        AvgEtchRate2=sum(LayerEtchRate)/length(LayerEtchRate)
        for(c=2:length(RelHeight))
            TimeEstimate2(c)=RelHeight(c)*1e6/AvgEtchRate2;
            fprintf('Pt #%d:%g um, TimeGuess2=%g hr,ActTime=%g hr from Etches %s\n',c,
RelHeight(c)*1e6, TimeEstimate2(c), ManualEtchTime(c), ManualEtchNames{c});
        end

        %outputvariables at the end are:
        ManualEtchTime'
        -RelHeight'*1e6
    end
end
```

Appendix R Detailed Fabrication Recipe for Quartz:SU-8 Testing Wings

Not all tested samples used exactly this recipe, specifically, this is the recipe for sample "QSU8_17a"

1. Pattern the metal contacts on the quartz top side
    a. Take sample from packaging
    b. Clean:
        i. Clean tweezers with DI water (DI), then soap, acetone, then methanol, sonicating (i.e., placing the beaker containing the wafer and solvent in an ultrasonic bath) for each
        ii. N2 gun, wafer
        iii. Rinse with DI water
        iv. Sonicate in DI water for 5 minutes
        v. Clean tweezers with IPA
        vi. Rinse with IPA
        vii. Sonicate in IPA for 5 minutes
    c. Dry with nitrogen, then 150^{o}C hotplate for 10 minutes
    d. 1808 for metallization of front-pattern with electrical isolation
        i. Coat with Shinetzu microprime
            1. coat using syringe
            2. let sit for 15s
            3. Spin at 4kRPM for 30s
        ii. Spin on 1808
            1. 1808 on surface using syringe
            2. let sit for 30s
            3. spin at 4kRPM for 30s
        iii. Let sit for 5 min in a dish, partially covered (8 min for _17)
        iv. Soft bake at 90^{o}C for 2 min
        v. Over expose, at least 40mJ/cm^2 (40mJ/5.2mW = 7.7s) (QSU8_17a: 21s @ 1.9mW)

1. Use the electrical contact mask shown in **Error! Reference source not found.**. 1808 is a positive photoresist, so the area of the photomask which allows light to pass = the area where 1808 will be removed = the area where metal will remain on the quartz after liftoff.



Appendix Figure R-1: Electrical contact mask for 1808

vi. Toluene dip, 5-6min (QSU8_17a: 5.5min)

vii. Blow dry with N2

viii. Dev in S351 (corrosives):DI water mixed at 1:5

1. Over dev: 3x normal time of 30s = 90s (or so; swirling)

a. (QSU8_17a: Developed for 2x 90s since the wafer was still reddish; unclear whether this was actually necessary).

ix. Rinse in DI, being careful to make pattern parallel to the water

x. Checked on alpha stepper that everything went well

e. Metallize: 25nm Cr, 150nm Au in the clean room.

f. Liftoff in acetone in ultrasonic immediately afterwards

i. rinse, then about 5min in ultrasonic

g. Then methanol in the ultrasonic

   i. rinse, dry, then proceed to hotplate:

2. Pattern the SU-8 on the top of the quartz as the passive layer:

   a. SU8 structural for front:

      i. Dehydration bake: 5 min at 150°C setting, under glass slide lid

      ii. UV-Ozone clean, ~~10~~ 5 min

      iii. Use syringe and filter, and disposable beaker.

      iv. Spin as per _5's recipe: 30s @ 3k after slow ramp up

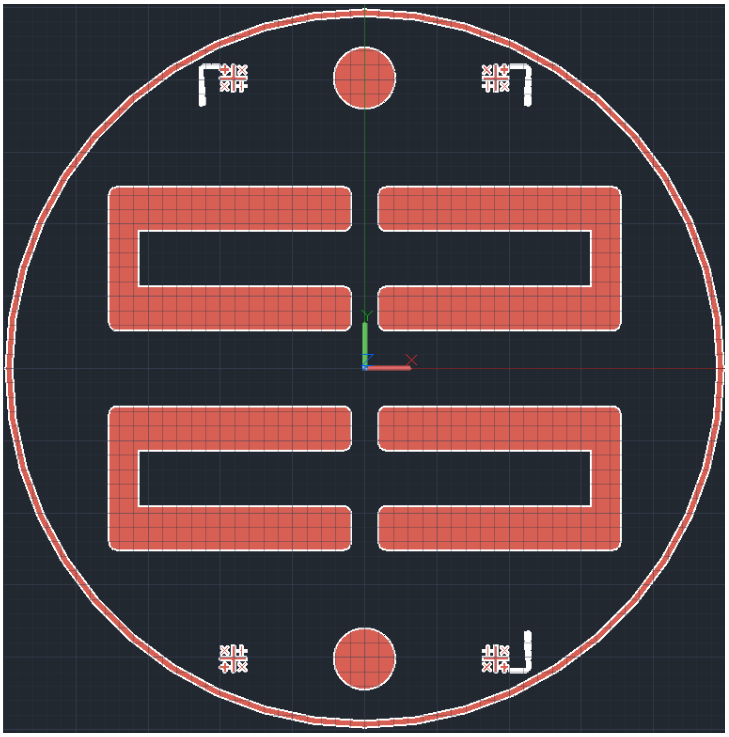
      v. 2.5min soft bake @107°C hot plate setting, on foil under glass lid

         1. (Put onto already hot hot plate then turn off after 2.5min and let cool for about ~~5~~ 2 min after letting out the hot air under the glass & replacing it)

      vi. 460mJ/cm^2 exposure = 88s @5.3mW (100s @4.1mW/cm^2)

         ~~1. 256s @ 1.8mW~~. 242s @1.9mW (4.04min)

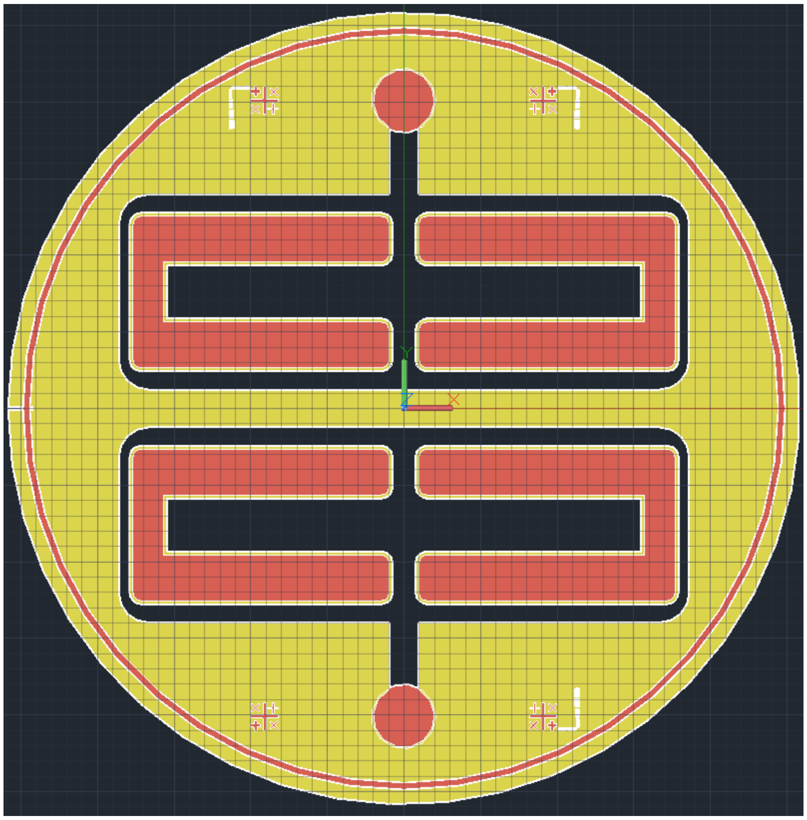
Apply the SU-8 front mask



Appendix Figure R-2: SU-8 Front mask

SU-8 is a negative photoresist, so the area of the mask which passes light through = the area where SU-8 is strengthened and will remain on the surface after development.

Here is the SU-8 mask overlaid on the metal mask:



Appendix Figure R-3: SU-8 mask overlaid on the metal mask

Notice that the metal mask blocks more area around the wings than the SU-8 mask does. Since the SU-8 mask around the wings is nominally coincident with the release mask (and therefore the ultimate wings themselves), this is so that the metal doesn't extend to the edges of the wings, in order to prevent high voltages from arcing through the air at the edges, since air has a lower breakdown field than quartz does. This would also work if the SU-8 covered the entire surface (except the circles at the top and bottom to allow access to the contact pads), but would then require a longer wing-release RIE to etch through the quartz and the SU-8.

vii. 3.5min PEB @ 107^{o}C HP setting (As per soft bake. Did 3.5 minutes for actual bake, then 9 minute cool).

viii. Develop, 2.5min, swirl during, then rinse in IPA, then blow dry. (Developed for 4.5 minutes)

ix. HB: 185^{o}C setting, 30 minutes.

1. Gradually rose temp to 185 over ~~6~~ 3 minutes, then left at temperature for ~~32~~ 27 minutes before turning off. Took off hot plate ~~5~~ (0 for _17, 3 for _18) min later.

b. and measure on alpha stepper & micrometer.

i. Alpha stepper measures:

1. _17: 7.9, 8, 7.6, 7.8
2. _18: 8, 7.8, 7.8

ii. Micrometer:

1. _17: 104um.

a. Etch depth measured as 96microns. So, if that's to be believed, then the quartz is 8microns thick at the end.

b. QSU8_17 was etched to 7.5 & 9.5 microns for the two different wings.

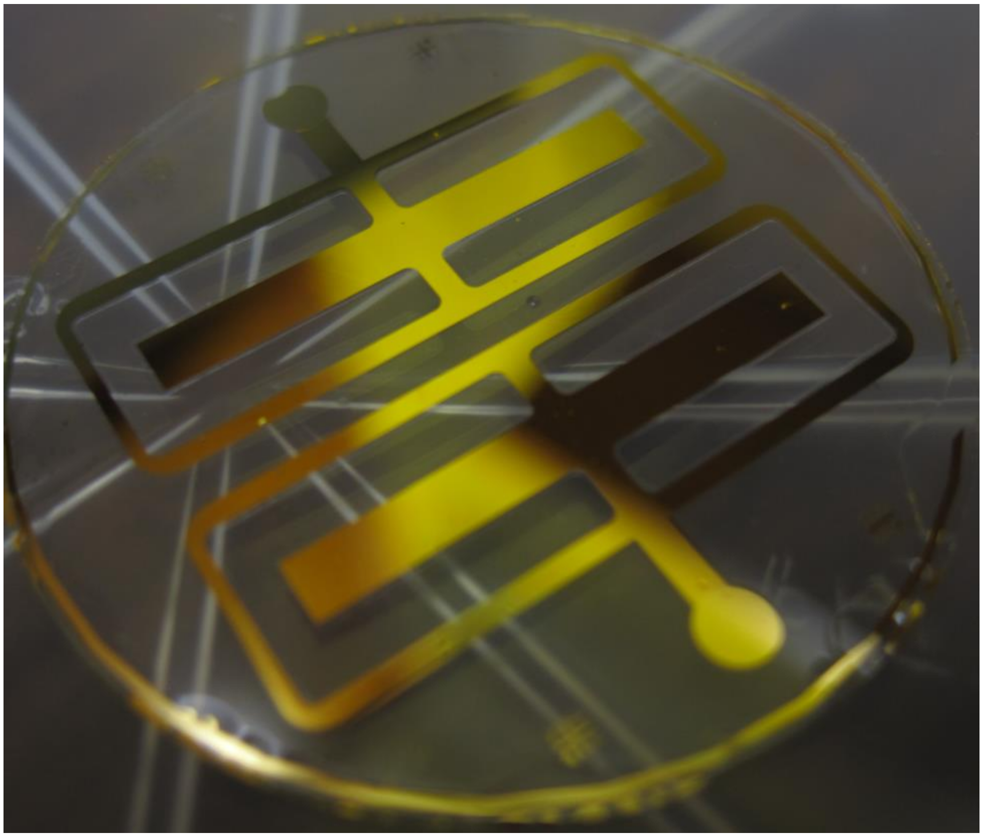
c. Optical profilometry found --> 7.513 um.

iii. Micrometer _18: 106-109um; 111-114 incl. the PR. (Measured 12/12/2011 2:46 PM). Wing thickness measured as 11-12um (in CCEM pictures)

1. If wing pictures thickness are to be believed, and SU8 is still 8um, then quartz is about 4um.
2. But there's some problems with this:

a. I've lost the CCEM pictures that made that measurement.

b. The alpha stepper and RIE predictions for the thinning etch done suggest a removal of 96.3µm. A 104um original wafer would make a thickness of 7.7um. 7.7um gives exactly the right kind of curvature in WA2 (as of 24/02/2012 1:50 PM) of about 83^o.

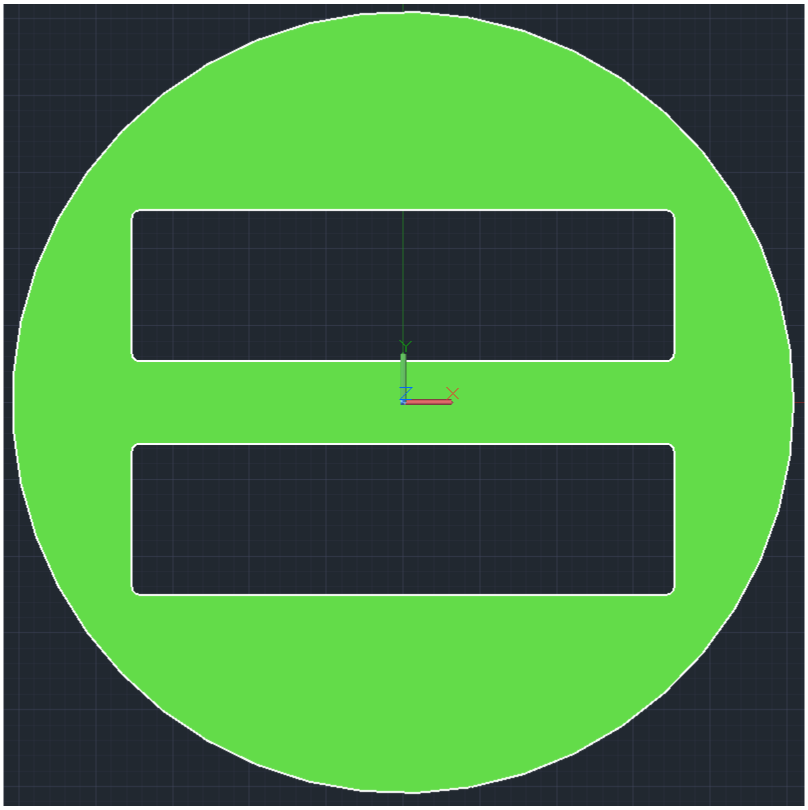


Appendix Figure R-4: Sample QSU8_14 after SU-8 patterning

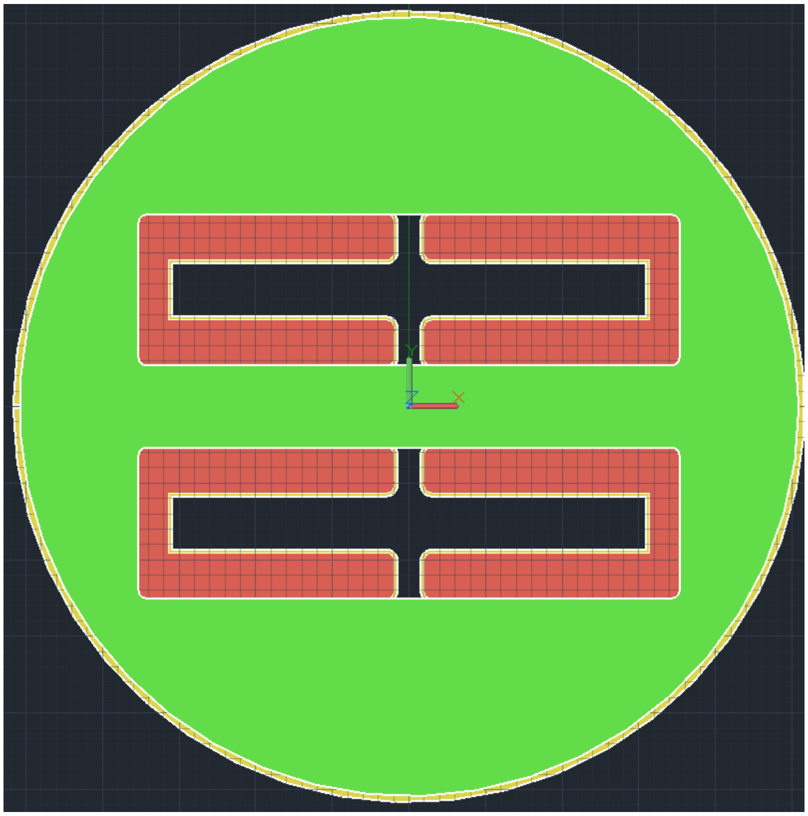
3. Thin the quartz from the bottom side with RIE

   a. RIE log recipe for QSU8_17, (90% Ar)

Date	Run Time	SF_6 flow rate	Ar flow rate	P	Pwr	V_{Bias}	V_{PP}	Load:	Tune:
	h	[sccm]	[sccm]	[mTorr]	[W]	[V]	[V]	[V]	[V]
2011-08-26	6.55	1.793	16.4	15	300	705	2110	40.8	60.6
2011-08-27	15.47	1.793	16.4	13	300	706	2110	40.8	60.9
2011-08-27	14.68	1.793	16.4	13	300	706	2110	40.8	60.9
2011-08-28	0.867	1.793	16.4	14	300	702	2100	41	60.9
Total:	37.57								

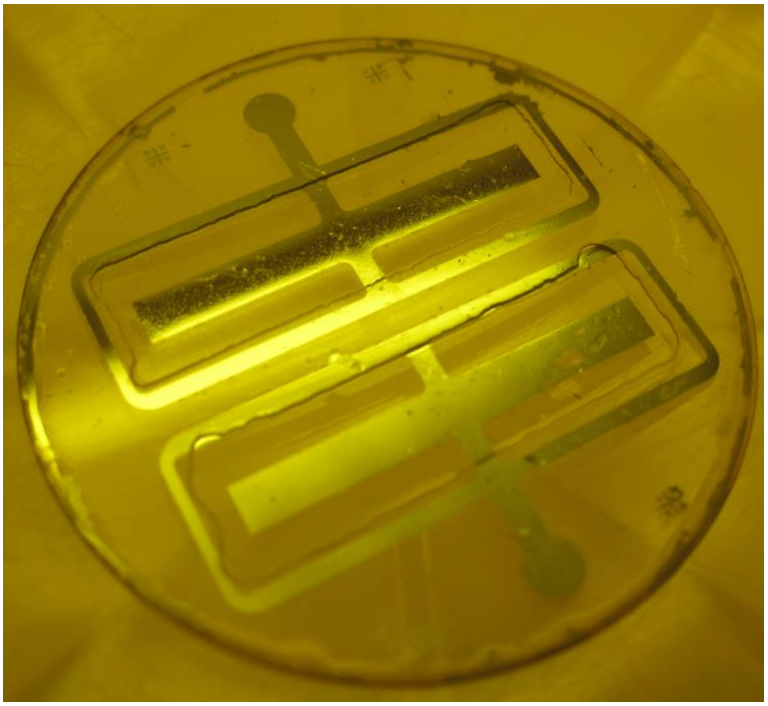


Appendix Figure R-5: Bottom mask



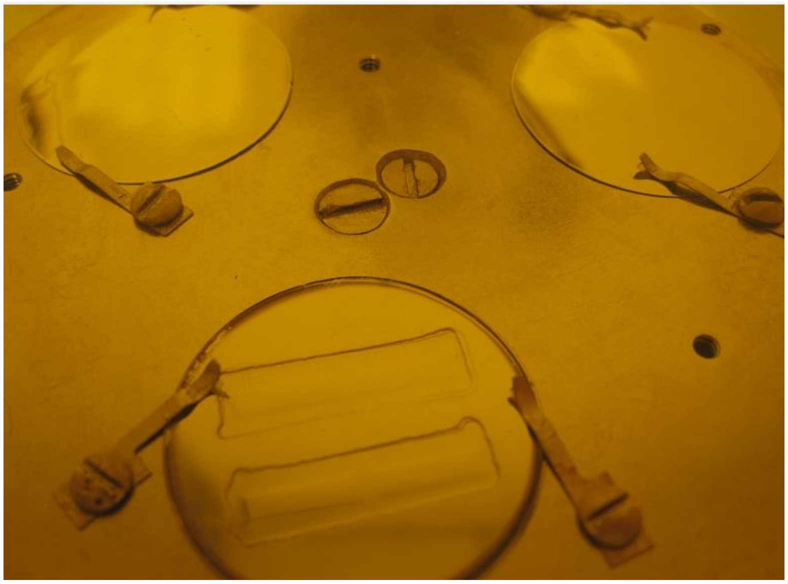
Appendix Figure R-6: Bottom mask overlaid on the metal mask

The quartz thinning mask is actually not a photomask but a shadow mask made from fused silica by milling this pattern by hand.



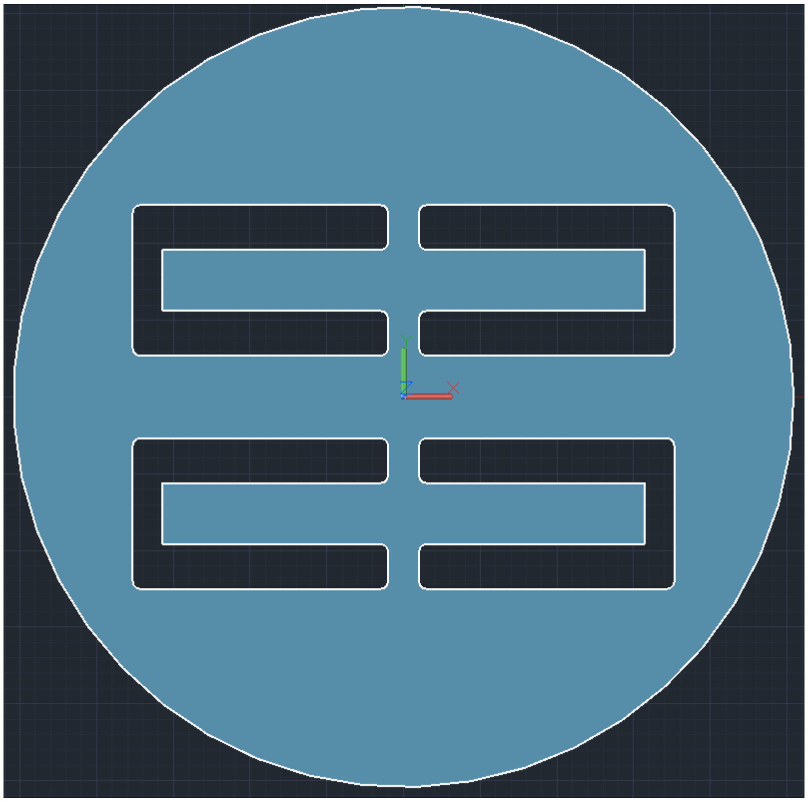
Appendix Figure R-7: The back of QSU8_17 prior to back-metallization illustrates the importance of proper mask alignment

4. Metallize the back side
   a. There is no mask for this step; rather, the back side is metallized uniformly in a low vacuum (300mTorr?) sputtering system to encourage coating the sidewalls of the etched quartz regions in order to make electrical contact to the bottoms of the wings.
   b. QSU8_17 back metallize: 25 August 2011, in clean room
      i. Thickness was 25nm Cr, 150nmAu.
   c. QSU8_18 back metallized on 12 Oct 2011 with QSU8_19's front metallization:
      i. Cr: 10nm, Au: 100nm



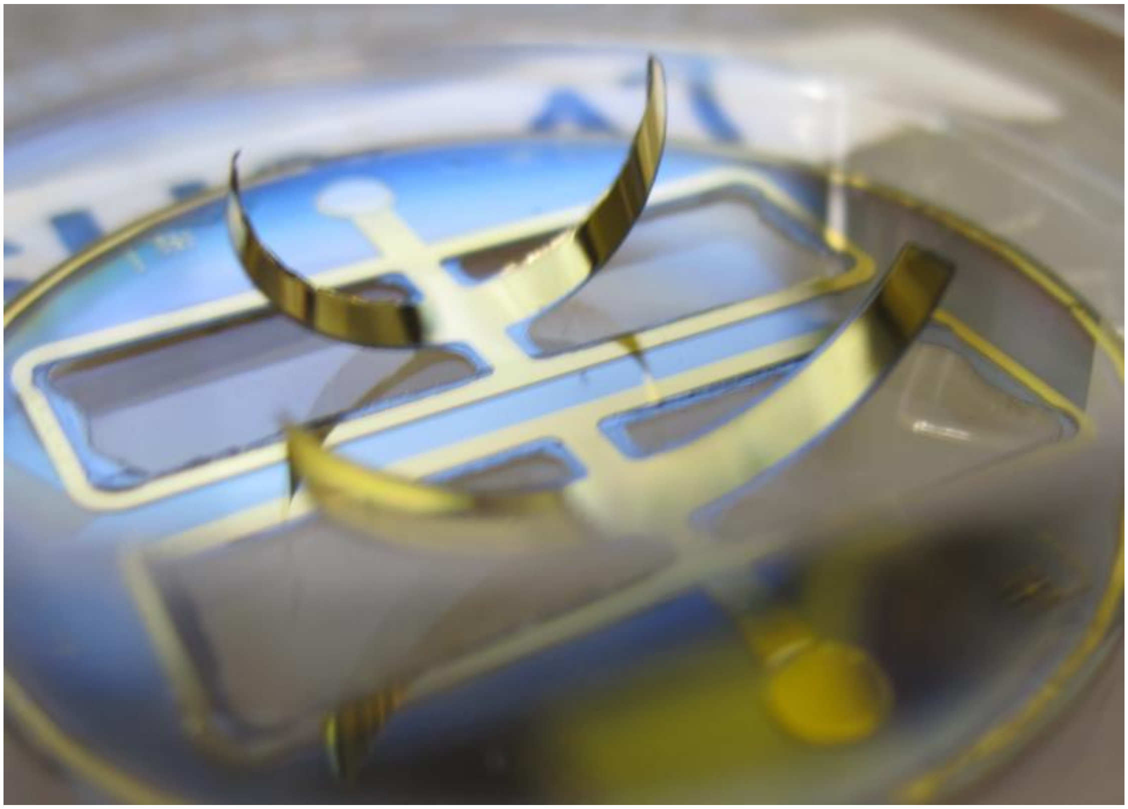
Appendix Figure R-8: After metallizing the back of QSU8_17 (bottom) and the fronts of QSU8_18 & 19

5. RIE release the wings



Appendix Figure R-9: Release mask

The release mask, Appendix Figure R-9, is a shadow mask made of nickel using waterjet cutting. As the release etch is a through-etch, there's no final surface so the surface quality is irrelevant and nickel shadow masking is preferable to quartz shadow masking for two reasons: ease of mask fabrication and longevity of mask in RIE.



Appendix Figure R-10: Sample QSU8_17 after release

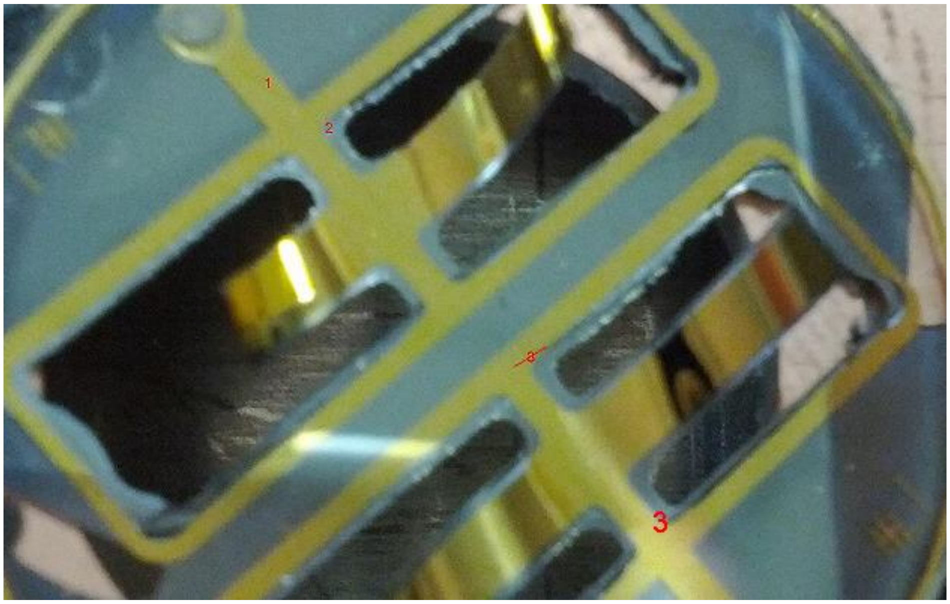
Appendix S Optical profilometry of QSU8_17a

Thickness measures of parts of the wings were accomplished several ways:

1. Contact profilometry,
2. Micrometry,
3. Measured etch rate multiplied by etch time, and
4. Optical profilometry.

The measured etch rate was reliant on the aggregate of many etches with the same parameters and so is typically less justified than an individual measurement, except where large errors are possible. Optical profillometry was the most reliable measurement technique, and also the least invasive. Optical profilometry was accomplished with a ZYGO New View profilometer. Unfortunately, the software supplied with the interferometer, MetroPro, was not sufficient for simultaneously correcting for curvature and tilt or averaging several specific measurements, so I wrote a separate MATLAB program to manually parse its data and accomplish this.

In this section I demonstrate this program by in turn examining 3 sections of the completed QSU8_17:

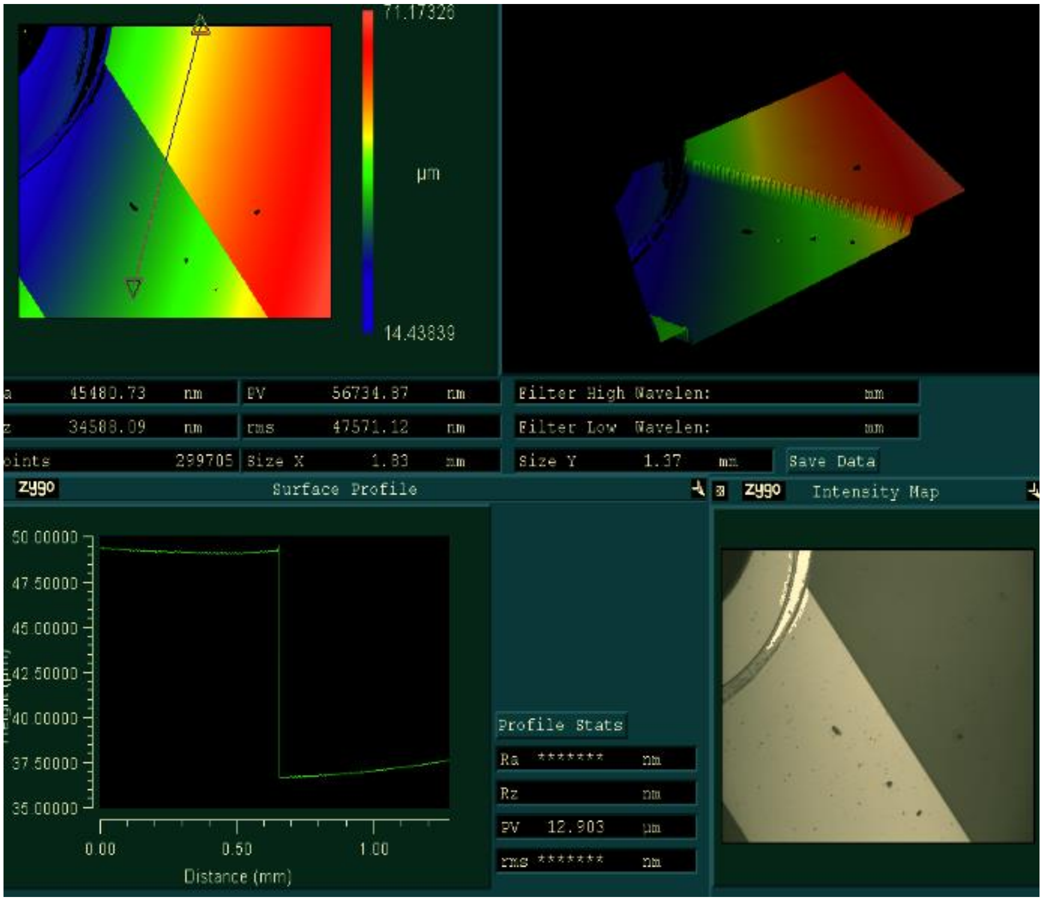


Appendix Figure S-1: Examined sections of QSU8_17 using optical profilometry

Section S.1 QSU8_17: Section #1

Section #1 is useful for determining the index of refraction of the SU-8; it does this by measuring a distance to gold through SU-8, another to gold through air, and a third to the SU-8 surface.

Here is the image of the processed data produced by MetroPro:



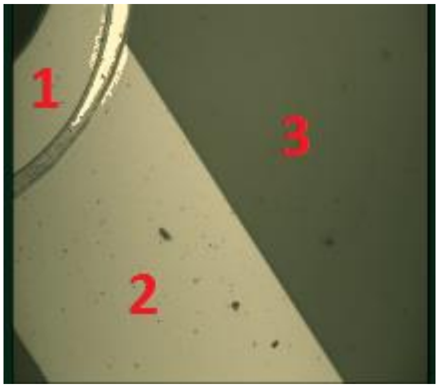
```
a        45480.73   nm | PV      56734.87   nm | Filter High Wavelen:        mm
z        34588.09   nm | rms     47571.12   nm | Filter Low  Wavelen:        mm
oints        299705    | Size X      1.83   mm | Size Y      1.37   mm
```

```
Profile Stats
Ra   *******   nm
Rz             nm
PV   12.903    µm
rms  *******   nm
```

Appendix Figure S-2: Measuring the SU-8 thickness with MetroPro

There are 3 areas of interest here:

1. Top left: Gold with no SU-8 on top of it.
2. Bottom leftish strip: gold with SU-8 on top of it
3. Top right: SU-8 directly on quartz.



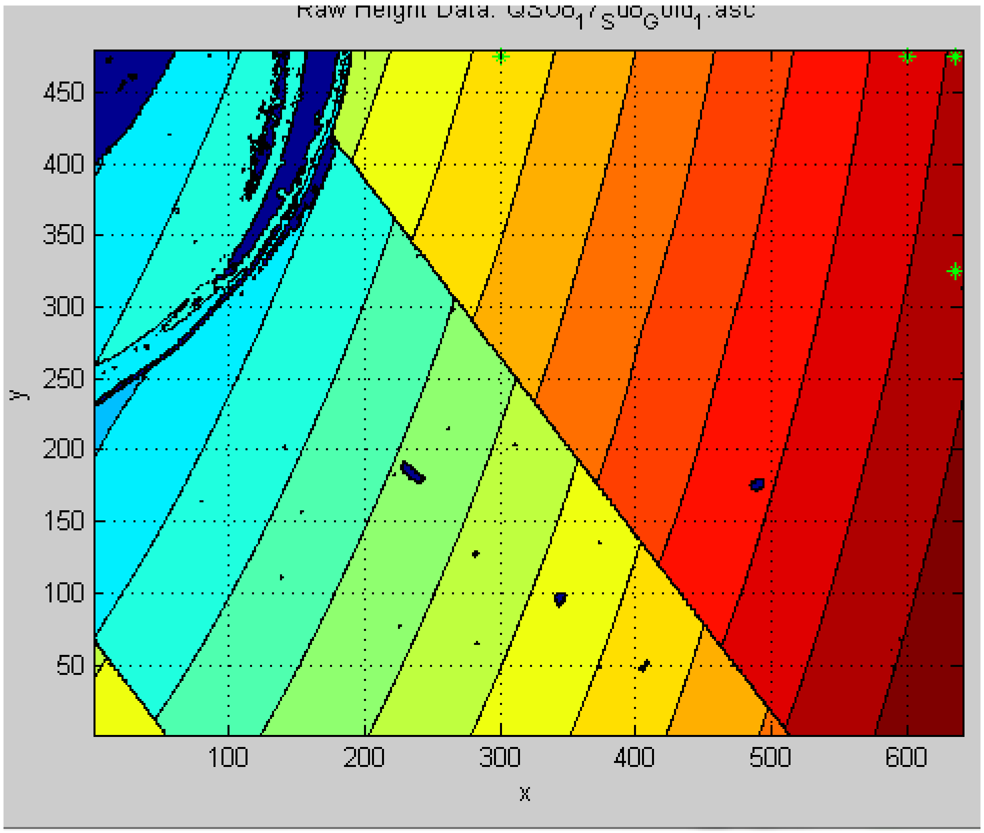
Appendix Figure S-3: Subsection labels

Because the thin gold surfaces in regions 1 & 2 are much more reflective than the quartz and SU-8 surfaces, we conveniently have a top-surface reflection at region 3 and a subsurface reflection at region 2. Therefore, the SU-8 height is the difference between 1 & 3, and the SU-8 index of refraction comes from the extra depth that region 2 appears compared to region 1 (once we know the SU-8 thickness).

To do these measurements, use this set of inputs for the ZygoProc program:

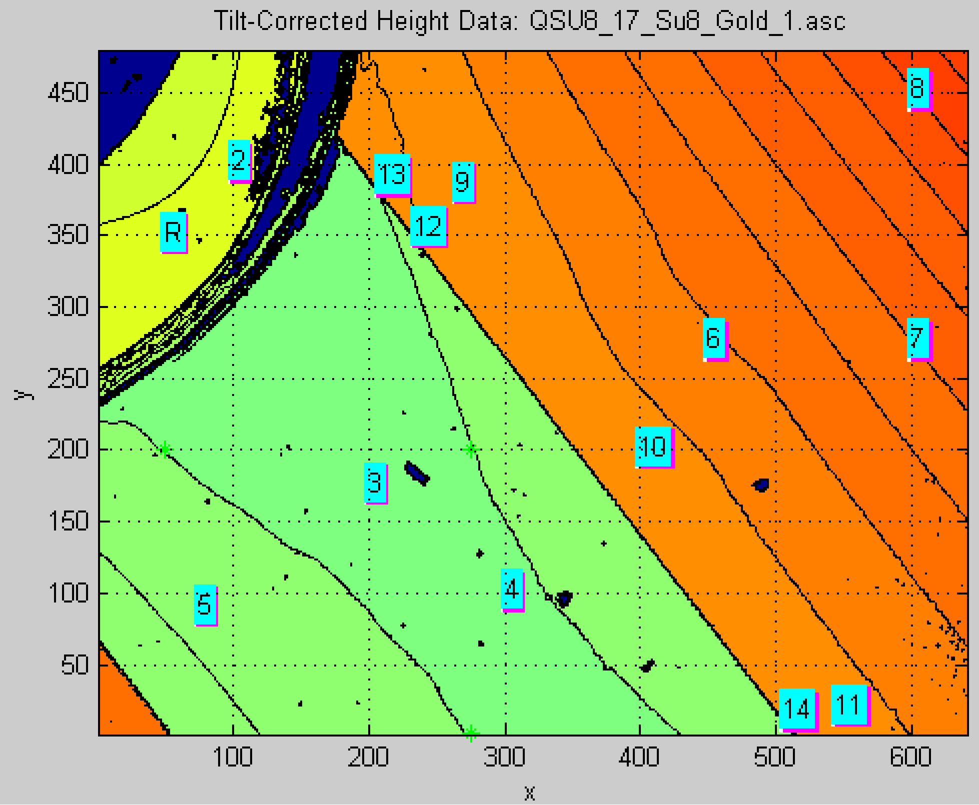
```
%ASCII file to read:
DatFileName='QSU8_17_Su8_Gold_1.asc';
Data3dPlot=0;
%Tilt correction points
RawDataPlot=0;
RawDataContourNumber=20;
xFitPoints=275+[-225 0 0 0];  %Dx move, then Dy move
yFitPoints=200+[0 0 0 -198]; %Dx move, then Dy move



%Points To Calculate Height At:
TiltCorrectedPlot=1;
TiltCorrectedContourNumber=100;
xHeightPoints=[50,100,200,300,75,450, 600, 600, 265, 400, 544, 234, 208, 506];
yHeightPoints=[350,400,175,100,90,275,275, 450, 385, 200, 20, 354, 390, 16];
xHeightPoints2=xHeightPoints+2;
yHeightPoints2=yHeightPoints;
xHeightPoints3=xHeightPoints;
yHeightPoints3=yHeightPoints+2;
EtchCalcs = 0;  %Turn off etch rate calculations if this is not an etch depth test
```

Zygoproc.m uses 2 pairs of points and does a two-axis tilt-correction to remove the slope in x and y. The points it used are marked with green asterisks in the figure. Here is the result of using part of the SU-8 surface for this tilt correction:



Appendix Figure S-4: Thickness measurements indicate the need for tilt-correction

The large number of contour lines indicates that the SU-8 region was not a good choice for image calibration. If instead we specify that the gold area is flat and we correct the tilt to that, then the height map is more believable:



Appendix Figure S-5: Sections measured after tilt-correction

(The green asterisks are very faint. We can tell wehre they are from the code presented at the start of the section:

```
xFitPoints=275+[-225 0 0 0];  %Dx move, then Dy move
yFitPoints=200+[0 0 0 -198]; %Dx move, then Dy move
```

which is equivalent to:

```
xFitPoints=[50,275,275,275];   %Dx move, then Dy move
yFitPoints=[200,200,200,2]; %Dx move, then Dy move
```

This specifies 4 points: (50,200), (275,200), (275,200), (275,2).

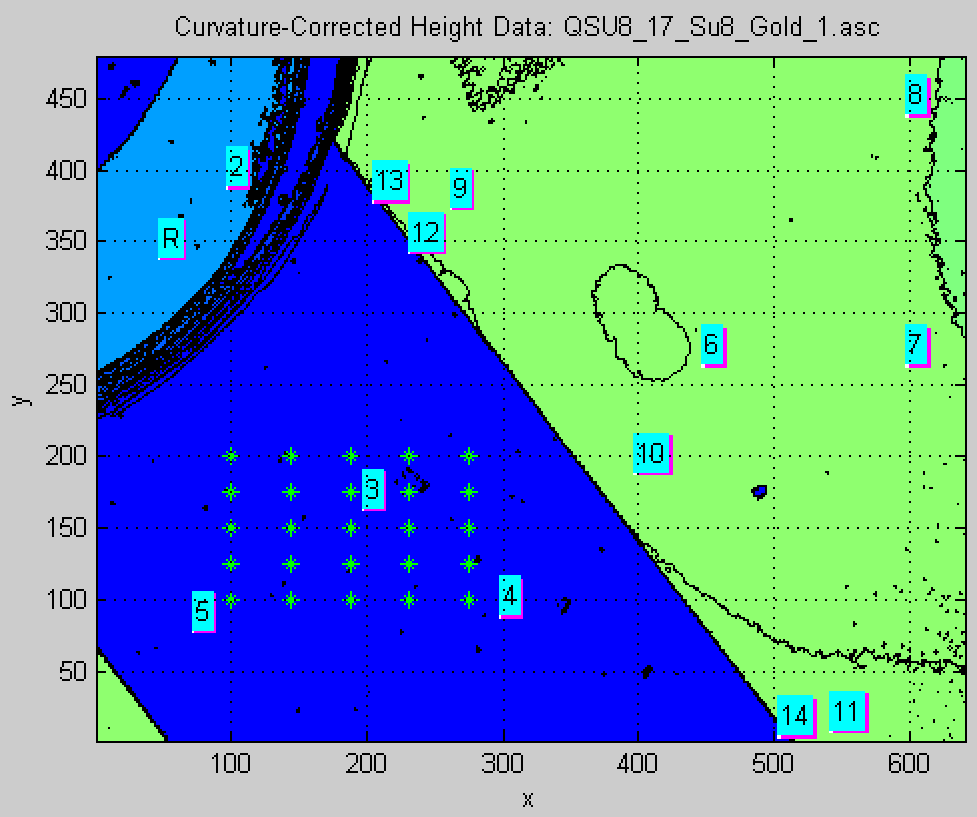
The first 2 are for tilt-correction in x, while the second 2 are for tilt-correction in y.)

Even better results come after doing a curvature correction:

ZygoProc_CurvatureRemoval2d_2.m with input:

```
%ASCII file to read:
DatFileName='QSU8_17_Su8_Gold_1.asc';
Data3dPlot=0;
```

```
%Tilt correction points
FitLowerLeft=[100,100];
FitUpperRight=[275,200];
NumberOfPointsPerFitLine = 5;

%Points To Calculate Height At:
TiltCorrectedPlot=1;
TiltCorrectedContourNumber=100;
xHeightPoints=[50,100,200,300,75,450, 600, 600, 265, 400, 544, 234, 208,
506];
yHeightPoints=[350,400,175,100,90,275,275, 450, 385, 200, 20, 354, 390,
16];
xHeightPoints2=xHeightPoints+2;
yHeightPoints2=yHeightPoints;
xHeightPoints3=xHeightPoints;
yHeightPoints3=yHeightPoints+2;
EtchCalcs = 0;  %Turn off etch rate calculations if this is not an etch
depth test
RawDataPlot=0;
RawDataContourNumber=20;
```



Appendix Figure S-6: Array of points correcting for tilt and ccurvature

This is from ZygoProc_CurvatureRemoval2d_2.m. Green asterisks define the points used to fit a second-order polynomial.

The reference point is marked with an 'R', point 2 is a check of the uniformity in this region (it should be the same height as the reference).

Points 3-5: apparent depth of the gold under the SU-8 compared to the reference point (the same gold layer with no SU-8 on top).

Points 6-8: far away SU-8

Points 9-11: close SU-8

Points 12-14: very close SU-8

The program outputs this chart:

```
Relative depths of each point, along with the average & stdev of all 3 colours
for that point:
```

P#	D1	D2	D3	Avg	StDev
#	um	um	um	um	um
2	-0.0177	-0.0164	-0.0190	-0.0177	0.0013
3	-4.9909	-4.9898	-4.9919	-4.9909	0.0011
4	-4.9873	-4.9866	-4.9885	-4.9874	0.0010
5	-4.9783	-4.9769	-4.9792	-4.9781	0.0012
6	7.5743	7.5638	7.5736	7.5706	0.0059
7	7.3857	7.3832	7.3896	7.3862	0.0032
8	7.3666	7.3641	7.3639	7.3649	0.0016
9	7.6037	7.6043	7.6057	7.6046	0.0010
10	7.5733	7.5700	7.5751	7.5728	0.0026
11	7.7200	7.7162	7.7165	7.7175	0.0021
12	7.6502	7.6478	7.6470	7.6483	0.0017
13	7.6413	7.6379	7.6350	7.6381	0.0032
14	7.7606	7.7600	7.7556	7.7587	0.0027

(Each measure is actually 3 points very close together. A high standard deviation here indicates a bad point was picked).

Referring to the average column, the SU-8 thickness is the average of P# 6-14:

$t_{SU8} = 7.58 \pm 0.13 \mu\text{m}$ (using 1-σ as the uncertainty, i.e., 68% confidence that $t_{SU8} = 7.6 \pm 0.13$ if we had a normal distribution).

And referring to the average of P# 3-5: $h_{3:5} = -4.985 \pm 0.0066 \mu\text{m}$

(Notice that the gold under the SU-8 appears *lower* than the reference area. In reality, they're the same height, and that gold just has SU-8 on top of it rather than air, giving it a longer optical path length.)

The reflection off of the gold under the SU8 has an increased one-way optical path length compared to air by $(n-1)t$, where *t* is the thickness of the SU8. Therefore, we'd expect that relative to the reference, the height of P# 3-5 should be:

$$h_{3:5} = -\left(n-1\right)t_{SU8}$$

$$\left(n-1\right) = \frac{-h_{3:5}}{t_{SU8}}$$

$$\therefore n = 1 + \frac{-h_{3:5}}{t_{SU8}}$$

therefore

$$n = 1 + \frac{4.98547 \pm 0.006615}{7.584633 \pm 0.133818}$$

$$n = 1.657 \pm 0.012$$

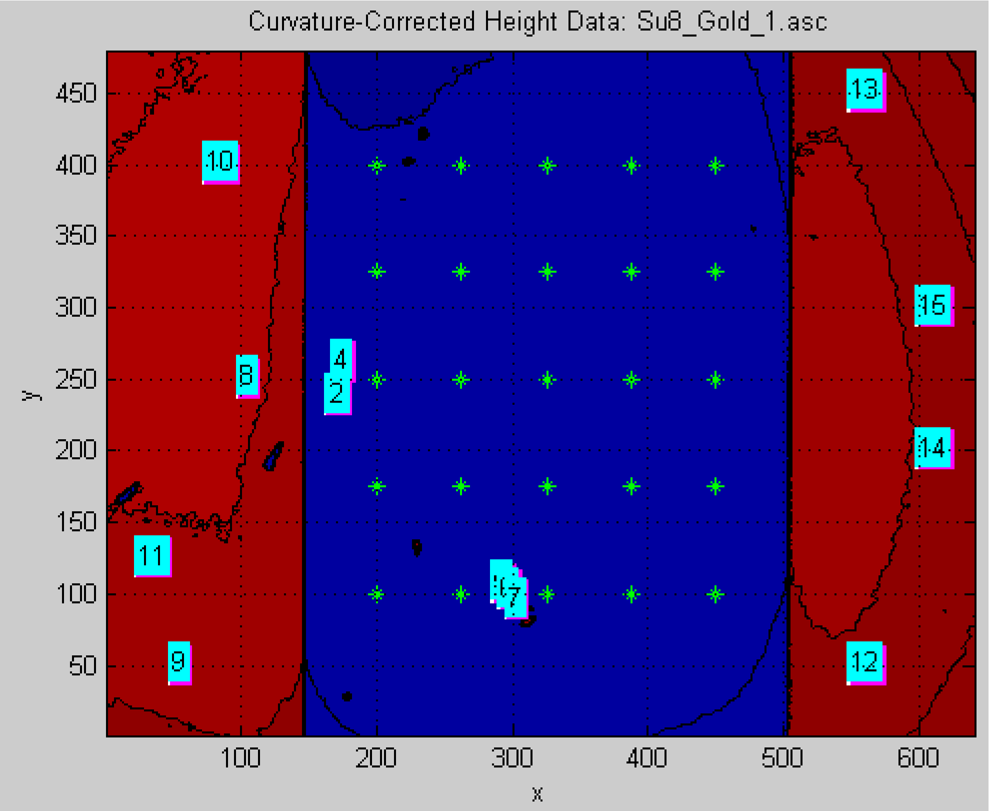
That's reasonable. Compare with the MEMS page on SU8 [81], SU-8's index of refraction is listed as between 1.575 and 1.8 depending on processing conditions. It's reasonable that SU-8 2000 would be in this range as well.

Section S.2 QSU8_17: Section #1, second scan

As a check, scan a part of the the gold-under-SU-8 farther away from the contact pad.

```
%ASCII file to read:
DatFileName='Su8_Gold_1.asc';
Data3dPlot=0;
%Tilt correction points
FitLowerLeft=[200,100];
FitUpperRight=[450,400];
NumberOfPointsPerFitLine = 5;

%Points To Calculate Height At:
TiltCorrectedPlot=1;
TiltCorrectedContourNumber=100;
xHeightPoints=[165,167,170,170,288,292,297,100, 50, 75, 25,    550, 550,
600, 600];
yHeightPoints=[238,238,261, 262, 107,102,95,250, 50, 400, 125, 050, 450,
200, 300];
xHeightPoints2=xHeightPoints+2;
```

```
yHeightPoints2=yHeightPoints;
xHeightPoints3=xHeightPoints;
yHeightPoints3=yHeightPoints+2;
EtchCalcs = 0;  %Turn off etch rate calculations if this is not an etch
depth test
RawDataPlot=0;
RawDataContourNumber=20;

PointsToAverage = 8:15;
```



Appendix Figure S-7: Curvature corrected gold bridge

Relative depths of each point, along with the average & stdev of all 3 colours for that point:

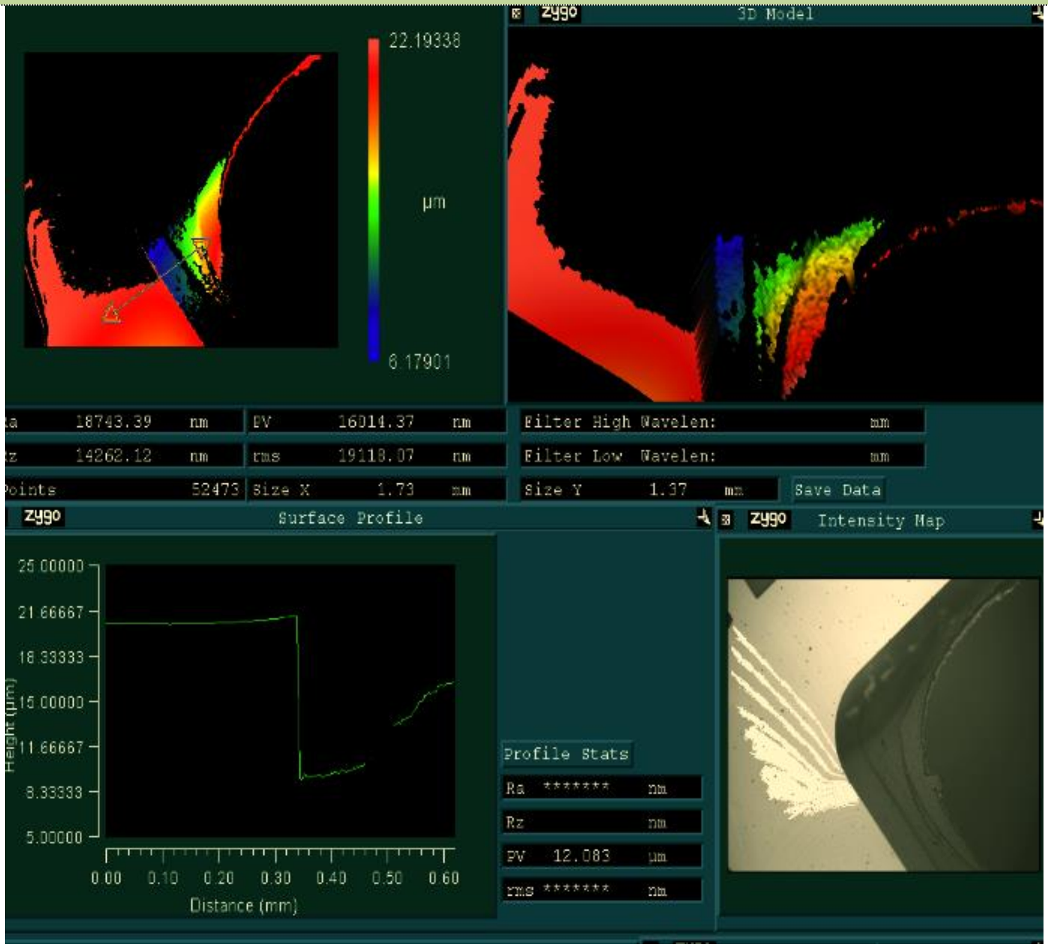
```
 P#       D1       D2       D3      Avg   StDev
  #       um       um       um       um      um
  2 -0.0009 -0.0026 -0.0004 -0.0013  0.0011
  3  0.0025  0.0011  0.0030  0.0022  0.0010
  4  0.0027  0.0016  0.0035  0.0026  0.0010
  5  0.0007 -0.0000  0.0000  0.0002  0.0004
  6  0.0019  0.0008  0.0017  0.0015  0.0006
  7  0.0022  0.0015  0.0005  0.0014  0.0009
  8 12.4831 12.4850 12.4853 12.4844  0.0012
  9 12.6114 12.6088 12.6086 12.6096  0.0016
 10 12.4278 12.4284 12.4289 12.4284  0.0006
 11 12.5443 12.5470 12.5466 12.5460  0.0014
 12 12.6821 12.6817 12.6798 12.6812  0.0013
 13 12.6980 12.7035 12.6984 12.7000  0.0030
 14 12.6695 12.6705 12.6689 12.6697  0.0008
 15 12.6868 12.6896 12.6862 12.6875  0.0018

Averaging these points:     2     3     4     5     6     7

find
mean = 0.00110084 um, stdev = 0.00144373 um

Averaging these points:     8     9    10    11    12    13    14    15

find
mean = 12.6008 um, stdev = 0.103452 um
```

Thus, the optical path one-way through the SU-8 is 12.6 ± 0.1 μm. Dividing by the index of refraction (found to be $n = 1.657 \pm 0.012$ in the previous section) gives the SU-8 thickness measure:

$h_{SU8} = 7.60 \pm 0.12$ μm.

So then, the SU-8 thickness here is effectively the same as measured right next to the contact pad.

Section S.3 QSU8_17: Section #2



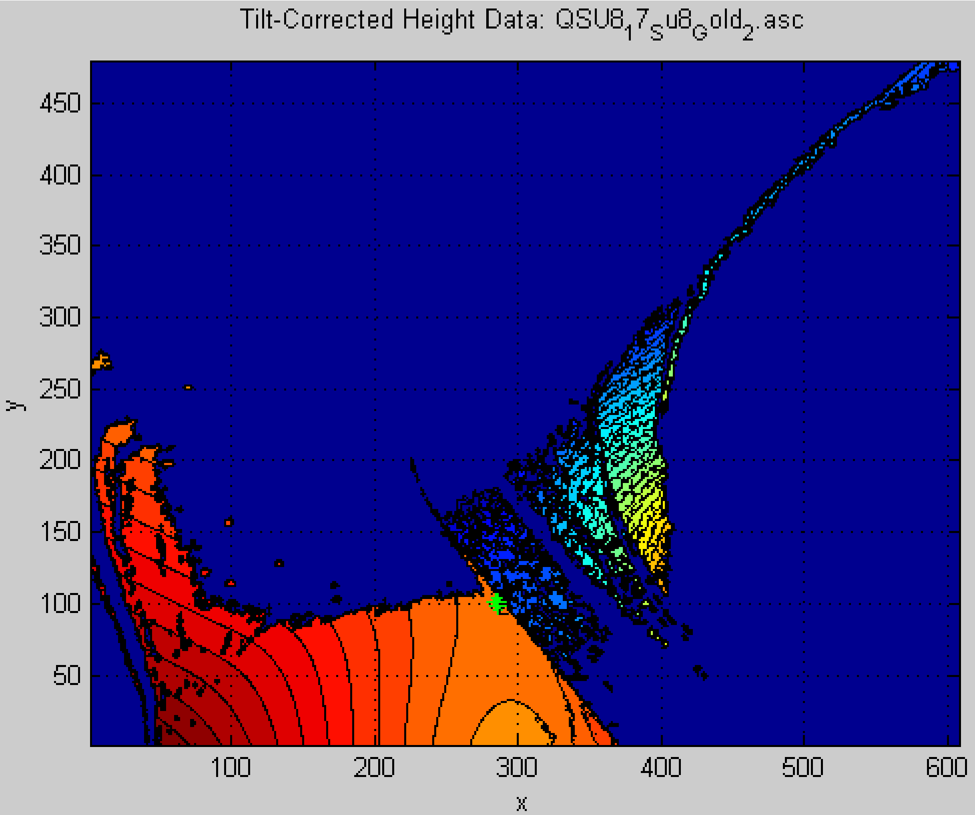
```
a        18743.39    nm   PV       16014.37    nm   Filter High Wavelen:           mm
z        14262.12    nm   rms      19118.07    nm   Filter Low  Wavelen:           mm
oints               52473 Size X       1.73    mm   Size Y        1.37    mm

Profile Stats
Ra  *******       nm
Rz                nm
PV   12.083       µm
rms *******       nm
```

Appendix Figure S-8: MetroPro output for section #2

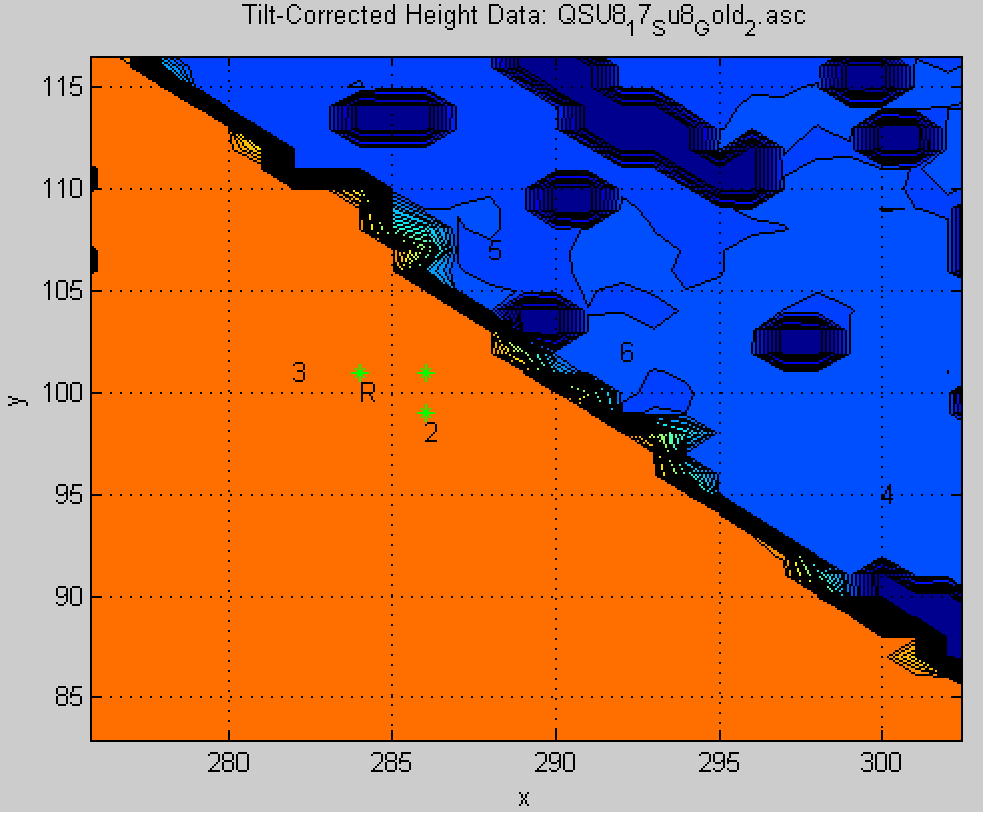
This one contains thickness info for the good pair of wings on QSU8_17. PV indicates a height of only 12 µm. And that's compared to the Gold surface! Can use the index of quartz to find the quartz thickness for this one.

```
%ASCII file to read:
DatFileName='QSU8_17_Su8_Gold_2.asc';
Data3dPlot=0;
%Tilt correction points
```

```
RawDataPlot=0;
RawDataContourNumber=20;
xFitPoints=[284,286,286,286];    %Dx move, then Dy move
yFitPoints=[101,101,101,99]; %Dx move, then Dy move

%Points To Calculate Height At:
TiltCorrectedPlot=1;
TiltCorrectedContourNumber=40;
xHeightPoints=[284,286,282,300,288,292];
yHeightPoints=[100,98,101, 95, 107,102];
EtchTimes=[5,13.5,6,8.5]; %Times of each etch, in hours
EtchNumberManualSpec=[0 1   1   9   9   11   11   15   15];
```



Appendix Figure S-9: Tilt-correction image reveals that the surface is very curved



Appendix Figure S-10: Close-up measure of the step depth nevertheless can determine thicknesses

Abs height at the reference point(284,100) = 1.59892e-005m

Relative height of point #2 @(286,98) = -0.00141724microns

Relative height of point #3 @(282,101) = 0.00177306microns

Relative height of point #4 @(300,95) = -11.5623microns

Relative height of point #5 @(288,107) = -11.91microns

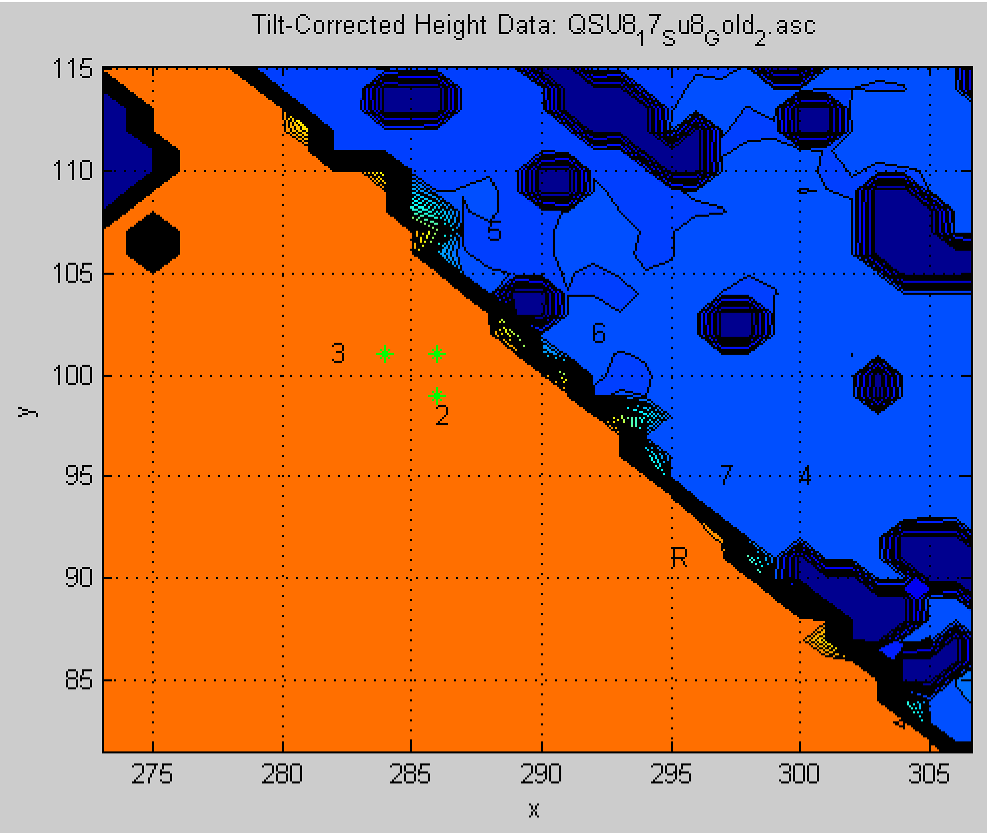
Relative height of point #6 @(292,102) = -11.7224microns

So, if positions 5 & 6 are most representative, being the closest, then the extra path length was about 11.65um, or (with n = 1.544), 7.5453µm.

Moving the ref closer to 4,

```
%ASCII file to read:
DatFileName='QSU8_17_Su8_Gold_2.asc';
Data3dPlot=0;
```

```
%Tilt correction points
RawDataPlot=0;
RawDataContourNumber=20;
xFitPoints=[284,286,286,286];    %Dx move, then Dy move
yFitPoints=[101,101,101,99]; %Dx move, then Dy move

%Points To Calculate Height At:
TiltCorrectedPlot=1;
TiltCorrectedContourNumber=40;
xHeightPoints=[295,286,282,300,288,292,297];
yHeightPoints=[91,98,101, 95, 107,102,95];
EtchTimes=[5,13.5,6,8.5]; %Times of each etch, in hours
EtchNumberManualSpec=[0 1   1   9   9   11  11  15  15];
```



Appendix Figure S-11: Another measure of step depth

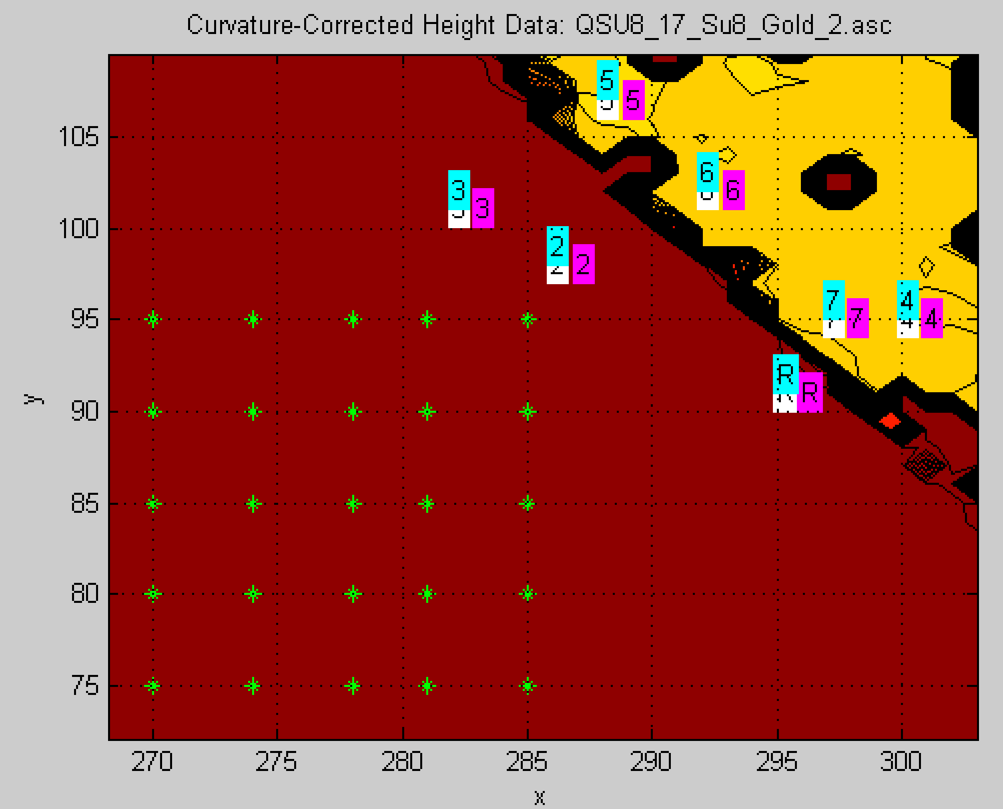
Abs height at the reference point(295,91) = 1.59961e-005m

```
Relative height of point #2 @(286,98) = -0.00823892microns
Relative height of point #3 @(282,101) = -0.00504863microns
Relative height of point #4 @(300,95) = -11.5691microns
Relative height of point #5 @(288,107) = -11.9168microns
Relative height of point #6 @(292,102) = -11.7292microns
Relative height of point #7 @(297,95) = -11.5973microns
```

So, maybe the height was closer to 11.5973. 11.6 gives 7.513µm.

Section S.3.1 Redo:

```
%ASCII file to read:
DatFileName='QSU8_17_Su8_Gold_2.asc';
Data3dPlot=0;
%Tilt correction points
FitLowerLeft=[270,75];
FitUpperRight=[285,95];
NumberOfPointsPerFitLine = 5;

%Points To Calculate Height At:
TiltCorrectedPlot=1;
TiltCorrectedContourNumber=100;
xHeightPoints=[295,286,282,300,288,292,297];
yHeightPoints=[91,98,101, 95, 107,102,95];
xHeightPoints2=xHeightPoints+1;
yHeightPoints2=yHeightPoints;
xHeightPoints3=xHeightPoints;
yHeightPoints3=yHeightPoints+1;
EtchCalcs = 0;  %Turn off etch rate calculations if this is not an etch
depth test
RawDataPlot=0;
RawDataContourNumber=20;

PointsToAverage = 4:7;
```



Appendix Figure S-12: Redoing the step depth after curvature correction as well

Output:

Averaging these points: 4 5 6 7

find

mean = -11.6967 um, stdev = 0.153815 um

Using the x-cut quartz index of refraction of n = 1.544, this means t_Q = -11.6967/1.544 = 7.575583:

$t_Q = 7.6 \pm 0.10$ μm.

Compared with:

Micrometer measure: original thickness = 104.

Etch depth measured with alpha stepper: 96 → t = 8 μm.

Etch depth calculated from calibrated etch rate and etch time: 96.2μm → t = 7.8 μm.

Appendix T Torsion Bar Theory & Setup

Section T.1 Spring Constant in Torsion

The twisting angle of a wire of length L_{wire} is directly proportional to the length and the torque applied and inversely proportional to the shear modulus ($G_{yz} = \frac{E_y}{2\left(1+\nu_{yz}\right)} \approx \frac{E_y}{2\left(1+0.33\right)}$) and torsion constant ($J = \frac{\pi r^4}{2}$):

$\theta = \frac{L_{wire}}{JG}\tau = -\frac{L_{wire}}{JG}\tau_{wire}$, where τ_{wire} is the torque applied *by* the wire to counter this torque τ *on* the wire [82].

Therefore, the "torsional spring constant" of the wire is $\kappa = \frac{JG}{L_{wire}} = \frac{\pi r_{wire}^4 G}{2L_{wire}}$, and $\tau = -\kappa\theta$.

Section T.2 Resonance calibration of spring constant

A "torsion bar" is a bar suspended at the end of a wire for measuring small forces by the angles they produce. The components that make up the spring constant of a coaxial wire used for suspension are difficult to measure directly, but fortunately an alternative is to measure the moment of inertia (by measuring the masses of components of the torsion bar and their location relative to the suspension point) and the natural resonant frequency of the bar.

Moment of Inertia:

When such a bar is suspended from a wire, the moment of inertia of the bar dominates:

$$I = I_{wire} + I_{bar} \approx I_{bar} = \frac{m_{bar}L_{bar}^2}{12} + m_{bar}\Delta x_{offCentreSuspension}^2 + m_1 x_1^2 + m_2 x_2^2$$

where $\Delta x_{offCentreSuspension}^2$ is how far off the centre of the bar the suspension wire is connected (applying the parallel axis theorem), and $m_i x_i^2$ is the MoI contributed by a point mass m_i located a distance x_i from the suspension point.

(e.g., for a bar suspended from the centre with equal masses m_{end} on each end:

$I_{bar} = \frac{m_{bar} L_{bar}^2}{12} + 2m_{end}\left(\frac{L_{bar}}{2}\right)^2$)

Resonant Frequency:

Newton's 2nd law says: $\tau = I\ddot{\theta}$. In the natural resonant case (no forcing), the torque on the bar is proved by the wire: $\tau = -\kappa\theta$, so that:

$-\kappa\theta = I_{bar}\ddot{\theta}$

This has the solution: $\theta = \theta_0 \sin\omega_{0TB}t$, where the natural angular frequency of the torsion balance is $\omega_{0TB} = \sqrt{\frac{\kappa}{I_{bar}}}$.

Therefore, $\kappa = I_{bar}\omega_{0TB}^2$.

Section T.3 Wing movement from Laser Spot Movement:

Account for piezo movement: technically, Δx_{spot} is the spot displacement *relative to the psd* (which is on a piezo stage). Since the psd is moving too, the spot movement relative to the table (left = positive) is

$$\Delta x_{spot0} = \Delta x_{spot} + \Delta x_{piezo},$$

where Δx_{piezo} is the psd movement relative to the table (due to its piezo actuators, left = positive)

These two displacements are found from:

1. $\Delta x_{spot} = \frac{\Delta V_{psd}}{\gamma_{psd}}$
2. $\Delta x_{piezo} = \Delta PiezoX_1 - \Delta PiezoX_2$ (The reason for the sign is in the piezo control part of the PTBC program:

```
self.c1 = -position_correction/2. #Higher axis 1 number = moves the stage to the left = moves the beam to the right = lower Vpsd
self.c2 = position_correction/2. #Higher axis 2 number = moves the stage
to the right = moves the beam to the left = higher Vpsd
```

Therefore:

1. Beam moves right = lower V_{psd}.
2. When wing moves right, beam moves left

--> This is why I define positive as *left* at the detector and *right* at the wing, so that:

$$\Delta x_{wing} = x_1 \Delta\theta$$

$$= x_1 \frac{\Delta x_{spot0}}{L_{beam}}$$

$$= x_1 \frac{\Delta x_{spot} + \Delta x_{piezo}}{L_{beam}}$$

$$= \frac{x_1}{L_{beam}} \left(\frac{\Delta V_{psd}}{\gamma_{psd}} + \Delta PiezoX_1 - \Delta PiezoX_2 \right)$$

Section T.3.1 Application: Cropping images by predicted wing position

Recipe for calculating the pixel shift of the on-pic:

Shift of on-pic to align it = position of *off* pic relative to *on* pic

$$\Delta x_{wing} = \frac{x_1}{L_{beam}} \left(\frac{\Delta V_{psd}}{\gamma_{psd}} + \Delta PiezoX_1 - \Delta PiezoX_2 \right) \times \gamma_{px/mm}$$

$$\Delta x_{wing} = \frac{0.28\text{m}}{0.854\text{m}} \left(\frac{\Delta V_{psd}}{2.151215\text{V}} + \frac{\Delta PiezoX_1 - \Delta PiezoX_2}{\text{mm}} \right) 26.2\text{px}$$

Where:

$$\Delta V_{psd} \equiv OffPicVpsd - OnPicVpsd$$

$$\Delta PiezoX_1 = OffPicPiezo1 - OnPicPiezo1$$

$$\Delta PiezoX_2 = OffPicPiezo2 - OnPicPiezo2$$

Section T.4 Force Measurement

Voltage from laser spot movement:

The angle is actually measured by the movement of a laser spot on a split photodiode. This puts out a voltage signal into the multimeter (after additional amplification and relative differencing) is linearly proportional to the displacement of the spot centre: $\Delta V_{spd} = \gamma_{spd}\Delta x_{spot}$ (and independent of the spot intensity, thanks to the relative differencing).

To determine γ_{spd}, we can piezoelectrically move the detector a known displacement and record the voltage change.

Laser spot movement from angle:

For a laser beam reflecting off of a mirror at the centre of the torsion bar and then travelling a distance L_{beam} before making a spot on a detector, the spot movement is (using the small angle approximation $\sin\theta \approx \theta \approx \tan\theta$) linearly related to the torsion angle: $\Delta x_{spot} = L_{beam}\Delta\theta$

Angle Measurement

Therefore, $\Delta\theta = \dfrac{\Delta x_{spot}}{L_{beam}} = \dfrac{\Delta V_{spd}}{L_{beam}\gamma_{spd}}$

Force Measurement

The force *F* produced by a wing at the end of the torsion bar generates a torque given by $\tau = x_1 F$ (where x_1 is the distance from the support to the end of the bar; for a symmetric bar-mass setup, $x_1 = L_{bar}/2$).

And the torque produces an equilibrium angle balanced by the restoring torque:

$x_1 F = \kappa\Delta\theta$

Therefore, the force is: $F = \dfrac{\kappa}{x_1}\Delta\theta = \dfrac{I_{bar}\omega_{0TB}^2}{x_1}\dfrac{\Delta V_{spd}}{L_{beam}\gamma_{spd}}$

(For convenience, define $F = \gamma_{FbyVspd}\Delta V_{spd}$)

Measurements required and how to do them:

1. I_{bar}:
    a. Measure mass of bar
    b. Measure mass of each mass at the ends of the bar
    c. Measure distance off-centre that the bar is supported at.
        i. $x_1 = \frac{L_{bar}}{2} + \Delta x_{offCentreSuspension}$
2. L_{beam}: use a meter stick
3. γ_{spd} : piezoelectrically move the detector a known distance and measure the change in the output voltage signal.
4. ω_{0TB} : sweep frequency of modulation of the wing signal and determine the resonant frequency of the bar.

Modulation of the Wing Signal

The signal applied to the wings when they are actuating is $V_{on}(t) = V_{amp}\sin(2\pi ft)$. ($f$ is not necessarily the wings' resonant frequency if doing a frequency response sweep)
The wings are not always actuating: In order to isolate the output signal from the wing force, the wing signal is modulated by a square wave of period T_{mod}:

$$V_{wing}(t) = square\left(\frac{t}{T}\right)V_{on}(t)$$

where $square(t) = \begin{cases} 1, & t \bmod 1 > 0.5 \\ 0, & t \bmod 1 < 0.5 \end{cases}$, and the modulo function means: $x \bmod y \equiv$ the remainder of x/y.

The output signal is

$$\Delta V_{spd}(t) = \frac{1}{\gamma_{FbyVspd}} F_{wing}\left(V_{wing}(t), f\right) = \frac{1}{\gamma_{FbyVspd}} F_{wing}\left(V_{amp}, f\right) square\left(\frac{t}{T} - \delta\right)$$

(the torsion bar cannot respond to the instantaneous force on the wing, only to the net force produced on average over many flap cycles. Hence, on the time-scale of the torsion bar's response, the force produced by the wings when they're being flapped depends on the frequency *f* and amplitude of the actuation voltage V_{amp}, but not on the instantaneous actuation voltage $V_{on}(t)$).

Therefore, the output is also be a square wave (possibly with a phase lag δ due to the torsion bar's response time). The force produced by an amplitude V_{amp} when the wing is on is $\gamma_{FbyVspd}$ times the *peak-to-peak* amplitude of ΔV_{spd} (or equivalently 2 times the amplitude of ΔV_{spd}).

<u>Demodulation of the output signal:</u>

Really, the output signal is due to the force, and noise:

$$\Delta V_{spd}(t) = \frac{F_{wing}(V_{amp}, f)}{\gamma_{FbyVspd}} square\left(\frac{t}{T} - \delta\right) + Noise(t)$$

A modulation-demodulation square wave filtering at the frequency of interest can eliminate the noise and reproduce the amplitude of the drive signal as follows:

$$V_{spdAmp} = v_{Filtered} = |v_{IPAvg}| + |v_{OPAvg}|$$

And has a phase of $\theta = \arctan(v_{OPAvg} / v_{IPAvg})$.

The peak-to-peak signal on the split photodiode is $\Delta V_{spd} = 2V_{spdAmp}$, and so therefore:

$$F_{wing}(V_{amp}, f) = 2\gamma_{FbyVspd} V_{spdAmp} = 2\gamma_{FbyVspd} v_{Filtered}$$

Section T.5 Summary:

The force output by the test structure is: $\boxed{F_{wing}(V_{amp}, f) = \gamma_{FbyVspd} 2v_{Filtered}}$

where

$\gamma_{FbyVspd} = \frac{I_{bar}\omega_{0TB}^2}{x_1} \frac{1}{L_{beam}\gamma_{spd}}$ is the force per voltage reading out of the circuits after the PSD: position sensitive detector.

$v_{Filtered} = v_{IPAvg} + v_{OPAvg}$ is the demodulated voltage reading

and $v_{IP}(t) \equiv \Delta V_{spd}(t) \times \left(2square\left(\frac{t}{T}\right) - 1\right)$, $v_{OP}(t) \equiv \Delta V_{spd}(t) \times \left(2square\left(\frac{t}{T} + 0.25\right) - 1\right)$

are the in-phase and -90^{o}-out-of-phase modulated voltage readings.

The components of the voltage to force gain are:

1. I$_{bar}$: moment of inertia of the torsion bar,

   $$I_{bar} = \frac{m_{bar} L_{bar}^2}{12} + m_{bar} \Delta x_{offCentreSuspension}^2 + m_1 x_1^2 + m_2 x_2^2$$

   a. Measure mass of bar m_{bar}

   b. Measure mass of each mass at the ends of the bar; m_1, m_2

   c. Measure distance off-centre that the bar is supported at: $\Delta x_{offCentreSuspension}$ (positive means the support is closer to the active arm)

      i. $x_1 = \frac{L_{bar}}{2} - \Delta x_{offCentreSuspension}$ (If the suspension is left of the bar's centre, that's a negative $\Delta x_{offCentreSuspension}$ and a larger x_1)

2. L_{beam}: distance from mirror to split-photodiode detector;

   a. measure with a meter stick

3. ω_{0TB}: torsion bar resonant frequency;

   a. sweep frequency of modulation of the wing signal and determine the resonant frequency of the bar.

   b. Or record the output with no input and fit a sine wave to it to determine the frequency (count peaks, divide by time)

4. γ_{spd}: rate of voltage signal change out of the split photodiode circuit with respect to laser spot position change; [V/m]

   a. piezoelectrically move the detector a known distance and measure the change in the output voltage signal voltage:

i. $\gamma_{spd} = \dfrac{\Delta V_{spd}}{\Delta x_{spot}}$

b. Modulate the piezo input signal with a peak-to-peak value of $\Delta x_{spotPP} = 2\Delta x_{spotAmp}$, and this leads (after processing as in the force measurement) to a demodulation signal of

i. $$v_{Filtered} = \gamma_{spd} \Delta x_{spotAmp}$$ $$\therefore \gamma_{spd} = \frac{v_{Filtered}}{\Delta x_{spotAmp}}$$.

Appendix U References

[1] "Nano Air Vehicle (NAV)." [Online]. Available: http://www.darpa.mil/Our_Work/DSO/Programs/Nano_Air_Vehicle_%28NAV%29.aspx. [Accessed: 16-Oct-2013].

[2] "Wasp Micro Air Vehicle (MAV) Developed by DARPA and AeroVironment," *Transcend*. [Online]. Available: http://www.transcend.ws/wasp-micro-air-vehicle-mav-developed-by-darpa-and-aerovironment/. [Accessed: 16-Oct-2013].

[3] "2011/11/24 TIME Magazine recognizes DARPA's Hummingbird Nano Air Vehicle." [Online]. Available: http://www.darpa.mil/newsevents/releases/2011/11/24.aspx. [Accessed: 16-Oct-2013].

[4] "Mean dry mass (male) - Fruit fly Drosophila melanogas - BNID 102570." [Online]. Available: http://bionumbers.hms.harvard.edu/bionumber.aspx?id=102570. [Accessed: 22-Oct-2013].

[5] "Mean dry mass (female) - Fruit fly Drosophila melanogas - BNID 102571." [Online]. Available: http://bionumbers.hms.harvard.edu/bionumber.aspx?s=n&id=102571&ver=2. [Accessed: 22-Oct-2013].

[6] "Flight, Reimagined: The First Robotic Hummingbird - 2011 Breakthrough Award Winner - Popular Mechanics." [Online]. Available: http://www.popularmechanics.com/technology/military/planes-uavs/flight-reimagined-the-first-robotic-hummingbird. [Accessed: 16-Oct-2013].

[7] "Black Hornet Nano," *Wikipedia, the free encyclopedia*. 23-Sep-2013.

[8] R. J. Wood, B. Finio, M. Karpelson, K. Ma, N. O. Pérez-Arancibia, P. S. Sreetharan, H. Tanaka, and J. P. Whitney, "Progress on 'pico' air vehicles," *Int. J. Robot. Res.*, vol. 31, no. 11, pp. 1292–1302, Sep. 2012.

[9] "Robotic insects make first controlled flight," *Harvard Gazette*. [Online]. Available: http://news.harvard.edu/gazette/story/2013/05/robotic-insects-make-first-controlled-flight/. [Accessed: 16-Oct-2013].

[10] "RoboBee," *Wikipedia, the free encyclopedia*. 14-Oct-2013.

[11] "Robot bee assembles in pop-up origami trick," *CNET*. [Online]. Available: http://news.cnet.com/8301-17938_105-57385893-1/robot-bee-assembles-in-pop-up-origami-trick/. [Accessed: 16-Oct-2013].

[12] J. R. Bronson, J. S. Pulskamp, R. G. Polcawich, C. M. Kroninger, and E. D. Wetzel, "PZT MEMS Actuated Flapping Wings for Insect-Inspired Robotics," in *IEEE 22nd International Conference on Micro Electro Mechanical Systems, 2009. MEMS 2009*, 2009, pp. 1047–1050.

[13] H.-Y. Chan, J. H. M. Lam, and W. J. Li, "A Biomimetic Flying Silicon Microchip: Feasibility Study," in *IEEE International Conference on Robotics and Biomimetics, 2004. ROBIO 2004*, 2004, pp. 447 –451.

[14] K. Mateti, R. A. Byrne-Dugan, C. D. Rahn, and S. A. Tadigadapa, "Monolithic SUEX Flapping Wing Mechanisms for Pico Air Vehicle Applications," *J. Microelectromechanical Syst.*, vol. 22, no. 3, pp. 527–535, 2013.

[15] "Forces on an Airplane." [Online]. Available: http://www.grc.nasa.gov/WWW/k-12/airplane/forces.html. [Accessed: 02-Sep-2013].

[16] "Eagle," *Wikipedia, the free encyclopedia*. 02-Sep-2013.

[17] "Naval Aviation Schools Command." [Online]. Available: http://www.netc.navy.mil/nascweb/sas/aero_dynamic_rollover.htm. [Accessed: 02-Sep-2013].

[18] "3 Essentials for Attracting Hummingbirds @ Jennifer Lyn KingJennifer Lyn King." [Online]. Available: http://www.jenniferlynking.com/2013/06/04/3-steps-to-attracting-hummingbirds/. [Accessed: 02-Sep-2013].

[19] "Helicopter rotor," *Wikipedia, the free encyclopedia*. 28-Aug-2013.

[20] "Pocket Rockets Pack a Punch | MIT Technology Review," *MIT Technology Review*. [Online]. Available: http://www.technologyreview.com/news/401025/pocket-rockets-pack-a-punch/. [Accessed: 02-Sep-2013].

[21] T. Dargent, X. Q. Bao, S. Grondel, G. L. Brun, J. B. Paquet, C. Soyer, and E. Cattan, “Micromachining of an SU-8 flapping-wing flying micro-electro-mechanical system,” *J. Micromechanics Microengineering*, vol. 19, no. 8, p. 085028, Aug. 2009.

[22] R. J. Wootton, “Functional Morphology of Insect Wings,” *Annu. Rev. Entomol.*, vol. 37, no. 1, pp. 113–140, 1992.

[23] M. Dickinson, “Solving the mystery of insect flight - Insects use a combination of aerodynamic effects to remain aloft,” *Sci. Am.*, vol. 284, no. 6, pp. 48–57, Jun. 2001.

[24] Y. Kubo, I. Shinoyama, T. Kaneda, and H. Miura, “Study on wings of flying microrobots,” in *, 1994 IEEE International Conference on Robotics and Automation, 1994. Proceedings*, 1994, pp. 834 –839 vol.1.

[25] “Euler–Bernoulli beam theory,” *Wikipedia, the free encyclopedia*. 17-Sep-2013.

[26] “Cantilever | ClipArt ETC.” [Online]. Available: http://etc.usf.edu/clipart/25700/25769/cantilever_25769.htm. [Accessed: 22-Oct-2013].

[27] S. Timoshenko, “Analysis of bi-metal thermostats,” *J. Opt. Soc. Am. 1917-1983*, vol. 11, p. 233, May 1925.

[28] D. L. DeVoe and A. P. Pisano, “Modeling and optimal design of piezoelectric cantilever microactuators,” *J. Microelectromechanical Syst.*, vol. 6, no. 3, pp. 266–270, Sep. 1997.

[29] D. Steinberg, *Vibration Analysis for Electronic Equipment*, 2nd ed. New York: Wiley, 1988.

[30] J. Yang, *An Introduction to the Theory of Piezoelectricity*. .

[31] G. Eason, “Wave propagation in a naturally curved elastic rod,” *J. Sound Vib.*, vol. 36, no. 4, pp. 491–511, Oct. 1974.

[32] M. Petyt, “Vibration of curved plates,” *J. Sound Vib.*, vol. 15, no. 3, pp. 381–395, Apr. 1971.

[33] T. Irie, G. Yamada, and K. Tanaka, “Natural Frequencies of In-Plane Vibration of Arcs,” *J. Appl. Mech.*, vol. 50, no. 2, pp. 449–452, Jun. 1983.

[34] J. R. BANERJEE, “EXPLICIT FREQUENCY EQUATION AND MODE SHAPES OF A CANTILEVER BEAM COUPLED IN BENDING AND TORSION,” *J. Sound Vib.*, vol. 224, no. 2, pp. 267–281, Jul. 1999.

[35] G. H. Zhang, Z. S. Liu, and H. H. Yoo, “In-plane vibration analysis of cantilevered circular arc beams undergoing rotational motion,” *J. Mech. Sci. Technol.*, vol. 22, no. 1, pp. 113–119, Jan. 2008.

[36] Landau and Lifshitz, *Mechanics, Third Edition: Volume 1 (Course of Theoretical Physics)*, 3rd ed. 1976.

[37] “Nonlinear resonance,” *Wikipedia, the free encyclopedia*. 16-Aug-2013.

[38] “Non-dimensionalization and scaling of the Navier–Stokes equations,” *Wikipedia, the free encyclopedia*. 31-Jul-2013.

[39] “Density of air,” *Wikipedia, the free encyclopedia*. 22-Oct-2013.

[40] M. Sun and S. L. Lan, “A computational study of the aerodynamic forces and power requirements of dragonfly (Aeschna juncea) hovering,” *J. Exp. Biol.*, vol. 207, no. Pt 11, pp. 1887–1901, May 2004.

[41] “Stokes flow,” *Wikipedia, the free encyclopedia*. 14-Aug-2013.

[42] M. A. Sleigh, “Mechanisms of flagellar propulsion,” *Protoplasma*, vol. 164, no. 1–3, pp. 45–53, Feb. 1991.

[43] F. O. Lehmann and m, “The control of wing kinematics and flight forces in fruit flies (Drosophila spp.).,” *J. Exp. Biol.*, vol. 201, no. 3, pp. 385–401, Feb. 1998.

[44] “Aerogel Composite Announces ENERGel(TM) Ultracapacitor | Business Wire,” 21-Feb-2006. [Online]. Available: http://www.businesswire.com/news/home/20060221006007/en/Aerogel-Composite-Announces-ENERGel-TM-Ultracapacitor. [Accessed: 05-Dec-2013].

[45] “IEEE Standard on Piezoelectricity,” *ANSIIEEE Std 176-1987*, p. 0_1–, 1988.

[46] “ATtiny4.” [Online]. Available: http://www.atmel.com/devices/ATTINY4.aspx?tab=documents. [Accessed: 20-Oct-2013].

[47] M. D. Minnick, G. A. Devenyi, and R. N. Kleiman, “Optimum reactive ion etching of x-cut quartz using SF6 and Ar,” *J. Micromechanics Microengineering*, vol. 23, no. 11, p. 117002, Nov. 2013.

[48] C. Hedlund, U. Lindberg, U. Bucht, and J. Soderkvist, “Anisotropic etching of Z-cut quartz,” *J. Micromechanics Microengineering*, vol. 3, no. 2, pp. 65–73, Jun. 1993.

[49] G. Thornell, H. Rapp, and K. Hjort, “X-cut miniature tuning forks realized by ion track lithography,” *Ultrason. Ferroelectr. Freq. Control IEEE Trans. On*, vol. 47, no. 1, pp. 8 –15, Jan. 2000.

[50] J. K. Vondeling, “Fluoride-based etchants for quartz,” *J. Mater. Sci.*, vol. 18, no. 1, pp. 304–314, 1983.

[51] C. T.-C. Nguyen, *Electrical Engineering C245: Introduction to MEMS Design - Lecture 7: Surface Micromaching*. UC Berkeley, 2011.

[52] H. Jansen, H. Gardeniers, M. de Boer, M. Elwenspoek, and J. Fluitman, “A survey on the reactive ion etching of silicon in microtechnology,” *J. Micromechanics Microengineering*, vol. 6, no. 1, pp. 14–28, Mar. 1996.

[53] A. Goyal, V. Hood, and S. Tadigadapa, “High speed anisotropic etching of Pyrex((R)) for microsystems applications,” *J. Non-Cryst. Solids*, vol. 352, no. 6–7, pp. 657–663, May 2006.

[54] L. Li, T. Abe, and M. Esashi, “Smooth surface glass etching by deep reactive ion etching with SF6 and Xe gases,” *J. Vac. Sci. Technol. B*, vol. 21, no. 6, pp. 2545–2549, Dec. 2003.

[55] P. Kao and S. Tadigadapa, “Micromachined quartz resonator based infrared detector array,” *Sens. Actuators Phys.*, vol. 149, no. 2, pp. 189–192, Feb. 2009.

[56] D. T. Chang, F. P. Stratton, R. L. Kubena, and R. J. Joyce, “Optimized DRIE etching of ultra-small quartz resonators,” in *Frequency Control Symposium and PDA Exhibition Jointly with the 17th European Frequency and Time Forum, 2003. Proceedings of the 2003 IEEE International*, 2003, pp. 829 – 832.

[57] H. Chen and C. Fu, “An investigation into the characteristics of deep reactive ion etching of quartz using SU-8 as a mask,” *J. Micromechanics Microengineering*, vol. 18, no. 10, p. 105001, Oct. 2008.

[58] D. Y. Choi, J. H. Lee, D. S. Kim, and S. T. Jung, “Formation of plasma induced surface damage in silica glass etching for optical waveguides,” *J. Appl. Phys.*, vol. 95, no. 12, pp. 8400–8407, Jun. 2004.

[59] D. Cheng, K. Sato, M. Shikida, A. Ono, K. Sato, K. Asaumi, and Y. Iriye, “Development of quartz etching database and 3-D micromachining simulation system,” in *Micromechatronics and Human Science, 2003. MHS 2003. Proceedings of 2003 International Symposium on*, 2003, pp. 281 – 285.

[60] E. Hein, D. Fox, and H. Fouckhardt, “Self-masking controlled by metallic seed layer during glass dry-etching for optically scattering surfaces,” *J. Appl. Phys.*, vol. 107, no. 3, p. 033301, 2010.

[61] X. H. Li, T. Abe, and M. Esashi, “Deep reactive ion etching of Pyrex glass using SF6 plasma,” *Sens. Actuators -Phys.*, vol. 87, no. 3, pp. 139–145, Jan. 2001.

[62] J. Lu, T. Ikehara, Y. Zhang, T. Mihara, T. Itoh, and R. Maeda, “High quality factor silicon cantilever driven by piezoelectric thin film actuator for resonant based mass detection,” *Microsyst Technol*, vol. 15, no. 8, pp. 1163–1169, Jun. 2009.

[63] S. Schmid and C. Hierold, “Damping mechanisms of single-clamped and prestressed double-clamped resonant polymer microbeams,” *J. Appl. Phys.*, vol. 104, no. 9, p. 093516, Nov. 2008.

[64] R. J. Wood, “The First Takeoff of a Biologically Inspired At-Scale Robotic Insect,” *IEEE Trans. Robot.*, vol. 24, no. 2, pp. 341–347, 2008.

[65] “Google Image Result for http://exploration.grc.nasa.gov/education/rocket/Images/shaped.gif.” [Online]. Available: http://www.google.ca/url?sa=i&source=images&cd=&cad=rja&docid=-Bk4EWFqjGgsCM&tbnid=5IG0RowkKHD1kM:&ved=0CAQQjxwwAA&url=http%3A%2F%2Fexploration.grc.nasa.gov%2Feducation%2Frocket%2Fshaped.html&ei=4P1WUtKDE-

amygGPu4GoDQ&psig=AFQjCNHPBZhwmlgo6tuNTcC4haU5hyPbKw&ust=1381519200401350. [Accessed: 10-Oct-2013].

[66] J. Soderkvist and K. Hjort, “The piezoelectric effect of GaAs used for resonators and resonant sensors,” *J. Micromechanics Microengineering*, vol. 4, no. 1, p. 28, Mar. 1994.

[67] “Viscosity,” *Wikipedia, the free encyclopedia*. 20-Oct-2013.

[68] E. W. Lemmon and R. T. Jacobsen, “Viscosity and Thermal Conductivity Equations for Nitrogen, Oxygen, Argon, and Air,” *Int. J. Thermophys.*, vol. 25, no. 1, pp. 21–69, Jan. 2004.

[69] “Vibration,” *Wikipedia, the free encyclopedia*. 15-Oct-2013.

[70] “Material Library.” [Online]. Available: http://www.comsol.com/products/material-library/. [Accessed: 09-Sep-2013].

[71] I. Roch, P. Bidaud, D. Collard, and L. Buchaillot, “Fabrication and characterization of an SU-8 gripper actuated by a shape memory alloy thin film,” *J. Micromechanics Microengineering*, vol. 13, no. 2, p. 330, Mar. 2003.

[72] “Periodic Table of the Elements by WebElements.” [Online]. Available: http://www.webelements.com/. [Accessed: 09-Sep-2013].

[73] “Chromium,” *Wikipedia, the free encyclopedia*. 08-Sep-2013.

[74] B. Li, M. Liu, and Q. Chen, “Low-stress ultra-thick SU-8 UV photolithography process for MEMS,” *J. MicroNanolithography MEMS MOEMS*, vol. 4, no. 4, pp. 043008–043008–6, 2005.

[75] L. Dellmann, S. Roth, C. Beuret, G.-A. Racine, H. Lorenz, M. Despont, P. Renaud, P. Vettiger, and N. F. de Rooij, “Fabrication process of high aspect ratio elastic and SU-8 structures for piezoelectric motor applications,” *Sens. Actuators Phys.*, vol. 70, no. 1–2, pp. 42–47, Oct. 1998.

[76] D. Bachmann, B. Schöberle, S. Kühne, Y. Leiner, and C. Hierold, “Fabrication and characterization of folded SU-8 suspensions for MEMS applications,” *Sens. Actuators Phys.*, vol. 130–131, pp. 379–386, Aug. 2006.

[77] H. Lorenz, M. Laudon, and P. Renaud, “Mechanical characterization of a new high-aspect-ratio near UV-photoresist,” *Microelectron. Eng.*, vol. 41–42, pp. 371–374, Mar. 1998.

[78] V. E. Bottom, “Measurement of the Piezoelectric Coefficient of Quartz Using the Fabry #x2010;Perot Dilatometer,” *J. Appl. Phys.*, vol. 41, no. 10, pp. 3941–3944, 1970.

[79] “Coaxial cable,” *Wikipedia, the free encyclopedia*. 19-Oct-2013.

[80] “Knudsen number,” *Wikipedia, the free encyclopedia*. 07-Aug-2013.

[81] C. J. Robin, A. Vishnoi, and K. N. Jonnalagadda, “Mechanical Behavior and Anisotropy of Spin-Coated SU-8 Thin Films for MEMS,” *J. Microelectromechanical Syst.*, pp. 1–1, 2013.

[82] “Torsion spring,” *Wikipedia, the free encyclopedia*. 05-Sep-2013.