A Parametric Test for Trend Based on Moving Order Statistics

A PARAMETRIC TEST FOR TREND BASED ON MOVING ORDER STATISTICS

BY

TAO TAN

A Thesis
Submitted to the School of Graduate Studies
in Partial Fulfilment of the Requirements
for the Degree of
Master of Science

McMaster University

Master of Science (2013)	McMaster University
(Statistics)	Hamilton, Ontario

TITLE:	A Parametric Test for Trend Based on Moving Order Statistics
AUTHOR:	Tao Tan
SUPERVISORS:	Professor N. Balakrishnan Professor A. Childs
NUMBER OF PAGES:	x, 70

Abstract

When researchers work on time series or sequence, certain fundamental questions will naturally arise. One of them will be whether the series or sequence exhibits a gradual trend over time. In this thesis, we propose a test statistic based on moving order statistics and establish an exact procedure to test for the presence of monotone trends. We show that the test statistic under the null hypothesis that there is no trend follows the closed skew normal distribution. An efficient algorithm is then developed to generate realizations from this null distribution. A simulation study is conducted to evaluate the proposed test under the alternative hypotheses with linear, logarithmic and quadratic trend functions. Finally, a practical example is provided to illustrate the proposed test procedure.

KEY WORDS: Parametric test, Moving order statistics, Time series; Monotone trends, Hypothesis, Closed skew normal distribution, Efficient algorithm, Simulation

Acknowledgements

I would like to express my deepest respect and sincere gratitude to my supervisor, Professor N. Balakrishnan, for his invaluable guidance, continuous support, constant encouragement and great patience.

I would also like to thank my co-surpervisor, Professor A. Childs, for his guidance, support, assistance and helpful suggestions.

I am grateful to Professor R. Viveros-Aguilera, for serving on my committee and for his insightful comments and feedback.

My thanks also go to Dr. William Volterman for his helpful discussions on my thesis.

Most importantly, I want to thank my parents, sister and husband for their endless love and support.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 What is Trend?

One of the main features of many time series or sequences is trend. Trend captures a slow and gradual change in a time series, which could be characterized by some property of the series over time. "Trend may be loosely defined as 'long-term change in the mean level' "(Chatfield, 2003). A key concept in traditional time series analysis is the decomposition of a series into trend, seasonal or periodic, and irregular components. Although such routine decomposition rarely occurs in modern analysis, a separate investigation of trend is still often needed in many practical situations (Meko, 2013). In particular, it is important to detect if there is a monotone trend in a sequence of observations.

1.2 What is the Typical Form of Trend?

We consider a model with a trend function

$$X_i = \beta t(i) + \varepsilon_i, \qquad i = 1, 2, ..., m, \quad \beta \in \mathbb{R}, \tag{1.1}$$

where $\varepsilon_i \sim i.i.d\ N(0,1)$ and $t(i)$ is a strictly increasing function. Then, $\beta = 0$ corresponds to the null hypothesis of no trend, and $\beta > 0$ ($\beta < 0$) marks an increasing (decreasing) trend in location. The trend function $t(i)$ can have different shapes. Linear, logarithmic and polynomial trends are the most commonly considered ones (Hofmann and Balakrishnan, 2006).

1.3 Problem of Interest

Our goal is to test for the presence of trend and determine if it is statistically significantly different from randomness. Thus, the hypotheses testing problem we are interested in is

$$H_0 : \beta = 0 \ \text{ vs. } \ H_1 : \beta > 0 \ \ (\beta < 0). \tag{1.2}$$

The null hypothesis means no trend or simply randomness, i.e., the hypothesis that $X_1, ..., X_m$ are independent and identically distributed (i.i.d.). The alternative hypothesis represents that there is an increasing (decreasing) trend. In this study, we establish an exact procedure to test the above hypotheses. In particular, we are interested in constructing a test statistic based on moving order statistics.

Chapter 2

The Closed Skew Normal Distribution

The closed skew normal (CSN) distribution is a superset of the normal family, which allows skewness features in the distribution. It was introduced by Domínguez-Molina *et al.* (2003) and González-Farías *et al.* (2004a) as a generalization of the multivariate skew normal distribution defined by Gupta *et al.* (2004). This distribution does preserve some important properties of the normal distribution. It is closed under full rank linear transformations, marginalization, sums, the conditional and joint distribution. Thus, it is more similar to the normal family than any other.

In this chapter, we introduce the definition of CSN distribution and discuss some of its properties and relationships to other distributions as well.

2.1 Definition of the CSN Distribution

In this section, we introduce the probability density function (pdf), cumulative distribution function (cdf) and the moment generating function (mgf) of the CSN distribution. The relevant proofs are given in Domínguez-Molina *et al.* (2003).

2.1.1 The pdf of the CSN Distribution

A p-dimensional random vector $\boldsymbol{Y}$ is said to have a CSN distribution, with parameters $\boldsymbol{\mu}, \Sigma, D, \boldsymbol{\nu}$ and Δ, if its pdf is of the form

$$f_{p,q}(\boldsymbol{y}; \boldsymbol{\mu}, \Sigma, D, \boldsymbol{\nu}, \Delta) = C\phi_p(\boldsymbol{y}; \boldsymbol{\mu}, \Sigma)\Phi_q(D(\boldsymbol{y} - \boldsymbol{\mu}); \boldsymbol{\nu}, \Delta), \quad \boldsymbol{y} \in \mathbb{R}^p, \tag{2.1}$$

with

$$C^{-1} = \Phi_q(\mathbf{0}; \boldsymbol{\nu}, \Delta + D\Sigma D'), \tag{2.2}$$

where $\phi_p(\cdot; \boldsymbol{\eta}, \Psi)$ and $\Phi_p(\cdot; \boldsymbol{\eta}, \Psi)$ are the pdf and cdf, respectively, of a p-dimensional normal distribution with mean vector $\boldsymbol{\eta}$ and covariance matrix Ψ, $p \geq 1$, $q \geq 1$, $\boldsymbol{\mu} \in \mathbb{R}^p$ and $\boldsymbol{\nu} \in \mathbb{R}^q$ are location parameters, $\Sigma \in \mathbb{R}^{p\times p}$ and $\Delta \in \mathbb{R}^{q\times q}$ are scale parameters, and $D \in \mathbb{R}^{q\times p}$ are skewness parameters. If $D = 0$, the pdf in (2.1) reduces to the usual multivariate normal one. We denote this distribution simply by $\boldsymbol{Y} \sim CSN_{p,q}(\boldsymbol{\mu}, \Sigma, D, \boldsymbol{\nu}, \Delta)$.

2.1.2 The cdf of the CSN Distribution

The cdf corresponding to the pdf in (2.1) is given by

$$F_{p,q}(\boldsymbol{y};\boldsymbol{\mu},\Sigma,D,\boldsymbol{\nu},\Delta) = C\Phi_{p+q}\left[\begin{pmatrix}\boldsymbol{y}\\ \mathbf{0}\end{pmatrix};\begin{pmatrix}\boldsymbol{\mu}\\ \boldsymbol{\nu}\end{pmatrix},\begin{pmatrix}\Sigma & -\Sigma D'\\ -D\Sigma & \Delta + D\Sigma D'\end{pmatrix}\right],$$

where C is as given in (2.2).

2.1.3 The mgf of the CSN Distribution

If $\boldsymbol{Y} \sim CSN_{p,q}(\boldsymbol{\mu},\Sigma,D,\boldsymbol{\nu},\Delta)$, then the mgf of $\boldsymbol{Y}$ is

$$M_{\boldsymbol{y}}(\boldsymbol{t}) = \frac{\Phi_q(D\Sigma\boldsymbol{t};\boldsymbol{\nu},\Delta + D\Sigma D')}{\Phi_q(\mathbf{0};\boldsymbol{\nu},\Delta + D\Sigma D')} e^{\boldsymbol{t}'\boldsymbol{\mu}+\frac{1}{2}\boldsymbol{t}'\Sigma\boldsymbol{t}}, \quad \boldsymbol{t} \in \mathbb{R}^p, \tag{2.3}$$

which is the product of the mgf of a p-dimensional Gaussian vector with mean $\boldsymbol{\mu}$ and covariance matrix Σ and the cdf of q dimensional normal distribution with mean $\boldsymbol{\nu}$ and covariance matrix $\Delta + D\Sigma D'$.

2.2 Construction of the CSN Distribution

In this section, we give a derivation of the CSN distribution based on a partitioned-conditional method due to Domínguez-Molina *et al.* (2003). This procedure is useful for simulating random vectors from this distribution.

2.2.1 Some Results on the Multivariate Normal Distribution

Let us first recall some results concerning the marginal and conditional distributions of the multivariate normal distribution.

Let $\boldsymbol{X} \sim N_{q+p}(\boldsymbol{\mu}, \Sigma)$. If we partition $\boldsymbol{X}$, its mean vector $\boldsymbol{\mu}$, and its covariance matrix Σ as

$$\boldsymbol{X}_{(q+p)\times 1} = \begin{pmatrix} \boldsymbol{X}_{1\ q\times 1} \\ \boldsymbol{X}_{2\ p\times 1} \end{pmatrix}, \quad \boldsymbol{\mu}_{(q+p)\times 1} = \begin{pmatrix} \boldsymbol{\mu}_{1\ q\times 1} \\ \boldsymbol{\mu}_{2\ p\times 1} \end{pmatrix},$$

and

$$\Sigma_{(q+p)\times(q+p)} = \begin{pmatrix} \Sigma_{11\ q\times q} & \Sigma_{12\ q\times p} \\ \Sigma_{21\ p\times q} & \Sigma_{22\ p\times p} \end{pmatrix},$$

then the following results are known:

1. the marginal distributions are $\boldsymbol{X}_1 \sim N_q(\boldsymbol{\mu}_1, \Sigma_{11})$ and $\boldsymbol{X}_2 \sim N_p(\boldsymbol{\mu}_2, \Sigma_{22})$;
2. the conditional distribution of $\boldsymbol{X}_1$, given $\boldsymbol{X}_2$, is

$$\boldsymbol{X}_1 \mid \boldsymbol{X}_2 = \boldsymbol{x}_2 \sim N_q(\boldsymbol{\mu}_1 + \Sigma_{12}\Sigma_{22}^{-1}(\boldsymbol{x}_2 - \boldsymbol{\mu}_2), \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}) \tag{2.4}$$

assuming Σ_{22} to be positive definite (Johnson and Wichern, 2007).

2.2.2 The Partitioned-Conditional Method

Domínguez-Molina *et al.* (2003) obtained the CSN distribution by considering some components of a normal random vector conditionally on other components being non-negative.

Consider two random vectors

$$\boldsymbol{W} = \boldsymbol{\mu} + \boldsymbol{E_1}$$
$$\boldsymbol{Z} = -\boldsymbol{\nu} + D\boldsymbol{E_1} + \boldsymbol{E_2},$$

where $\boldsymbol{E_1} \sim N_p(\mathbf{0}, \Sigma)$ and $\boldsymbol{E_2} \sim N_q(\mathbf{0}, \Delta)$ are independent random vectors, $D(q \times p)$ is an arbitrary matrix, $\boldsymbol{\mu} \in \mathbb{R}^p$, $\boldsymbol{\nu} \in \mathbb{R}^q$ and $\Delta(q \times q) > 0$.

Consider the joint distribution of $\boldsymbol{Z}$ and $\boldsymbol{W}$. Clearly,

$$\begin{pmatrix} \boldsymbol{Z} \\ \boldsymbol{W} \end{pmatrix} \sim N_{q+p}\left[\begin{pmatrix} -\boldsymbol{\nu} \\ \boldsymbol{\mu} \end{pmatrix}, \begin{pmatrix} \Delta + D\Sigma D' & D\Sigma \\ \Sigma D' & \Sigma \end{pmatrix}\right].$$

Then, Bayes theorem readily yields

$$f_{\boldsymbol{W}|\boldsymbol{Z}\geq\mathbf{0}}(\boldsymbol{w}|\boldsymbol{Z} \geq \mathbf{0}) = \frac{f_{\boldsymbol{W}}(\boldsymbol{w})}{P(\boldsymbol{Z} \geq \mathbf{0})} P(\boldsymbol{Z} \geq \mathbf{0}|\boldsymbol{W} = \boldsymbol{w}).$$

By using (2.4), the conditional distribution of $\boldsymbol{Z}$, given $\boldsymbol{W} = \boldsymbol{w}$, is obtained to be

$$\boldsymbol{Z}|(\boldsymbol{W} = \boldsymbol{w}) \sim N_q(-\boldsymbol{\nu} + D(\boldsymbol{w} - \boldsymbol{\mu}), \Delta).$$

Thus,

$$\begin{aligned} f_{\boldsymbol{W}|\boldsymbol{Z}\geq\mathbf{0}}(\boldsymbol{w}|\boldsymbol{Z} \geq \mathbf{0}) &= \frac{\phi_p(\boldsymbol{w}; \boldsymbol{\mu}, \Sigma)}{P(-\boldsymbol{Z} < \mathbf{0})} P(-\boldsymbol{Z} < \mathbf{0}|\boldsymbol{W} = \boldsymbol{w}) \\ &= \frac{\phi_p(\boldsymbol{w}; \boldsymbol{\mu}, \Sigma)}{\Phi_q(\mathbf{0}; \boldsymbol{\nu}, \Delta + D\Sigma D')} \Phi_q(\mathbf{0}; \boldsymbol{\nu} - D(\boldsymbol{w} - \boldsymbol{\mu}), \Delta) \\ &= \frac{\phi_p(\boldsymbol{w}; \boldsymbol{\mu}, \Sigma)}{\Phi_q(\mathbf{0}; \boldsymbol{\nu}, \Delta + D\Sigma D')} \Phi_q(D(\boldsymbol{w} - \boldsymbol{\mu}); \boldsymbol{\nu}, \Delta) \end{aligned}$$

which is the same as in (2.1).

2.3 Some Properties of the CSN Distribution

2.3.1 Moments of the CSN Distribution

Domínguez-Molina *et al.* (2003) obtained the first and second moments of the CSN distribution by taking the derivatives of the mgf in (2.3) at $\mathbf{0}$.

Let $\boldsymbol{Y} \sim CSN_{p,q}(\boldsymbol{\mu}, \Sigma, D, \boldsymbol{\nu}, \Delta)$. Then, the mean and variance of $\boldsymbol{Y}$ are given by

$$E(\boldsymbol{Y}) = \boldsymbol{\mu} + \Sigma D' \frac{[\nabla_s \Phi_q(\boldsymbol{s}; \boldsymbol{\nu}, \Delta + D\Sigma D')]'}{\Phi_q(\mathbf{0}; \boldsymbol{\nu}, \Delta + D\Sigma D')}\bigg|_{\boldsymbol{s}=\mathbf{0}}$$

and

$$\begin{aligned} V(\boldsymbol{Y}) = \Sigma &+ \Sigma D' \frac{\nabla_s \nabla_s' \Phi_q(\boldsymbol{s}; \boldsymbol{\nu}, \Delta + D\Sigma D')}{\Phi_q(\mathbf{0}; \boldsymbol{\nu}, \Delta + D\Sigma D')}\bigg|_{\boldsymbol{s}=\mathbf{0}} D\Sigma \\ &- \Sigma D' \frac{[\nabla_s \Phi_q(\boldsymbol{s}; \boldsymbol{\nu}, \Delta + D\Sigma D')]'}{\Phi_q(\mathbf{0}; \boldsymbol{\nu}, \Delta + D\Sigma D')}\bigg|_{\boldsymbol{s}=\mathbf{0}} \left\{ \frac{[\nabla_s \Phi_q(\boldsymbol{s}; \boldsymbol{\nu}, \Delta + D\Sigma D')]'}{\Phi_q(\mathbf{0}; \boldsymbol{\nu}, \Delta + D\Sigma D')}\bigg|_{\boldsymbol{s}=\mathbf{0}} \right\}' D\Sigma, \end{aligned}$$

where

$$\nabla_{\boldsymbol{s}} = \left(\frac{\partial}{\partial s_1}, \frac{\partial}{\partial s_2}, \ldots, \frac{\partial}{\partial s_q} \right)'$$

is the gradient operator.

2.3.2 Linear Transformations

The CSN distribution has some desirable properties under linear tranformations. For example, it is closed under translations, scalar multiplications and full row rank linear transformations (González-Farías *et al.*, 2004a). Specifically, if

$\boldsymbol{Y} \sim CSN_{p,q}(\boldsymbol{\mu}, \Sigma, D, \boldsymbol{\nu}, \Delta)$, then

1. for an arbitrary vector $\boldsymbol{b} \in \mathbb{R}^p$, $\boldsymbol{Y} + \boldsymbol{b} \sim CSN_{p,q}(\boldsymbol{\mu} + \boldsymbol{b}, \Sigma, D, \boldsymbol{\nu}, \Delta)$;
2. for a constant $c \in \mathbb{R}$, $c\boldsymbol{Y} \sim CSN_{p,q}(c\boldsymbol{\mu}, \Sigma c^2, Dc^{-1}, \boldsymbol{\nu}, \Delta)$;
3. for a matrix $A \in \mathbb{R}^{n\times p}(n \le p)$ of rank n,

$$A\boldsymbol{Y} \sim CSN_{n,q}(\boldsymbol{\mu}_A, \Sigma_A, D_A, \boldsymbol{\nu}, \Delta_A),$$

where $\boldsymbol{\mu}_A = A\boldsymbol{\mu}$, $\Sigma_A = A\Sigma A'$, $D_A = D\Sigma A'{\Sigma_A}^{-1}$ and $\Delta_A = \Delta + D\Sigma D' - D\Sigma A'{\Sigma_A}^{-1}A\Sigma D'$;

4. for an arbitrary vector $\boldsymbol{a} \in \mathbb{R}^p(\boldsymbol{a} \neq \boldsymbol{0})$,

$$\boldsymbol{a}'\boldsymbol{Y} \sim CSN_{1,q}(\mu_{\boldsymbol{a}}, \Sigma_{\boldsymbol{a}}, D_{\boldsymbol{a}}, \boldsymbol{\nu}, \Delta_{\boldsymbol{a}}),$$

where $\mu_{\boldsymbol{a}} = \boldsymbol{a}'\boldsymbol{\mu}$, $\Sigma_{\boldsymbol{a}} = \boldsymbol{a}'\Sigma\boldsymbol{a}$, $D_{\boldsymbol{a}} = D\Sigma\boldsymbol{a}{\Sigma_{\boldsymbol{a}}}^{-1}$ and $\Delta_{\boldsymbol{a}} = \Delta + D\Sigma D' - D\Sigma\boldsymbol{a}\boldsymbol{a}'\Sigma D'{\Sigma_{\boldsymbol{a}}}^{-1}$.

2.4 Connections with Other Distributions

The CSN distribution is a superset of the skew normal family. The univariate and multivariate skew normal distributions can be expressed as special cases of the CSN distribution.

2.4.1 Connection with Univariate Skew Normal Distribution

The univariate skew normal (SN) distribution was introduced by Azzalini (1985) . Let $\lambda, z \in \mathbb{R}$, $\phi(\cdot)$ and $\Phi(\cdot)$ be the pdf and cdf of $N(0, 1)$. We denote $Z \sim SN(\lambda)$ if its pdf is

$$\begin{aligned}\phi(z;\lambda) &= 2\phi(z)\Phi(\lambda z)\\ &= 2\phi(z;0,1)\Phi(\lambda z;0,1);\end{aligned}$$

that is, $Z \sim CSN_{1,1}(0,1,\lambda,0,1)$.

2.4.2 Connection with Multivariate Skew Normal Distribution

Azzalini and Dalla Valle (1996) proposed the multivariate version of the above SN distribution by conditioning on one random variable being positive. Let $\boldsymbol{\alpha},\boldsymbol{z} \in \mathbb{R}^k$, $\Omega \in \mathbb{R}^{k\times k}$ a positive definite matrix, and $\phi_k(\boldsymbol{z};\Omega)$ be the k-dimensional normal density with zero mean and covariance matrix Ω. Then, a k dimensional random vector $\boldsymbol{Z}$ follows the multivariate skew normal distribution according to Azzalini and Dalla Valle (1996), if its density function is of the form

$$\begin{aligned}f_k(\boldsymbol{z}) &= 2\phi_k(\boldsymbol{z};\Omega)\Phi(\boldsymbol{\alpha}'\boldsymbol{z})\\ &= 2\phi_k(\boldsymbol{z};\mathbf{0},\Omega)\Phi(\boldsymbol{\alpha}'\boldsymbol{z};0,1);\end{aligned}$$

that is, $\boldsymbol{Z} \sim CSN_{k,1}(\mathbf{0},\Omega,\boldsymbol{\alpha}',0,1)$.

Gupta *et al.* (2004) obtained a multivariate skew normal distribution by conditioning on the same number of random variables being positive. Let $\boldsymbol{\mu},\boldsymbol{Y} \in \mathbb{R}^p$, $\Sigma(p\times p) > 0$, $D(p\times p)$ be an arbitrary matrix, and $\phi_p(\cdot;\boldsymbol{\eta},\Psi)$ and $\Phi_p(\cdot;\boldsymbol{\eta},\Psi)$ denote the p.d.f. and the c.d.f. of p dimensional normal distribution with mean $\boldsymbol{\mu}$ and covariance matrix Σ. Then, a random vector $\boldsymbol{Y}(p \times 1)$ is distributed as $CSN_{p,p}$, according to Gupta *et al.* (2004), if its pdf is given by

$$\begin{aligned} f_p(\boldsymbol{y}; \boldsymbol{\mu}, \Sigma, D) &= \frac{1}{\Phi_p(\mathbf{0}; I + D\Sigma D')} \phi_p(\boldsymbol{y}; \boldsymbol{\mu}, \Sigma) \Phi_p(D(\boldsymbol{y} - \boldsymbol{\mu})) \\ &= \frac{1}{\Phi_p(\mathbf{0}; \mathbf{0}, I + D\Sigma D')} \phi_p(\boldsymbol{y}; \boldsymbol{\mu}, \Sigma) \Phi_p(D(\boldsymbol{y} - \boldsymbol{\mu}); \mathbf{0}, I_p); \end{aligned}$$

that is, $\boldsymbol{Y} \sim CSN_{p,p}(\boldsymbol{y}, \boldsymbol{\mu}, \Sigma, D, \mathbf{0}, I_p)$.

Compared to the multivariate skew normal distribution, the CSN distribution is a closed family as it contains its conditional densities by including an additional parameter $\boldsymbol{\nu}$, marginal densities by adding an extra parameter Δ, and the sum and joint distribution of independent CSN random vectors by introducing $\Phi_q(\cdot)$ for $q \geq 1$ among its members (González-Farías *et al.*, 2004b).

Chapter 3

Moving Order Statistics and a Test Procedure

3.1 Order Statistics

Order statistics and functions of order statistics play an important role in many fields of both statistical theory and practice. In this section, we introduce briefly the pdf, cdf and Markov property of order statistics.

3.1.1 The pdf of Order Statistics

Let $X_1, X_2, ..., X_n$ be independent, absolutely continuous random variables with common pdf $f(x)$ and cdf $F(x)$, and let $X_{1:n} \leq X_{2:n} \cdots \leq X_{n:n}$ denote the order statistics obtained by arranging the n random variables in a nondecreasing order of magnitude.

Considering that there are $n!$ equally likely orderings of the X_i's, by the multinomial method (Balakrishnan and Cohen, 1991), we could derive the joint density of all n order statistics to be

$$f_{X_{1:n},\ldots,X_{n:n}}(x_1,\ldots,x_n) = n!\prod_{i=1}^{n} f(x_i), \quad -\infty < x_1 < \cdots < x_n < \infty. \tag{3.1}$$

From (3.1), upon integrating out the variables

$$(X_{1:n},\ldots,X_{i-1:n}),(X_{i+1:n},\ldots,X_{n:n})$$

or by the binomial method, we could derive the marginal density function of $X_{i:n}$ as

$$f_{X_{i:n}}(x) = \frac{n!}{(i-1)!(n-i)!} f(x)\{F(x)\}^{i-1}\{1-F(x)\}^{n-i}, -\infty < x < \infty. \tag{3.2}$$

Similarly, from (3.1), upon integrating out the variables

$$(X_{1:n},\ldots,X_{i-1:n}),(X_{i+1:n},\ldots,X_{j-1:n}),(X_{j+1:n},\ldots,X_{n:n})$$

or by the multinomial method (Balakrishnan and Cohen, 1991), we could obtain the joint pdf of $X_{i:n}$ and $X_{j:n}$ $(1 \le i < j \le n)$ as

$$\begin{aligned} f_{X_{i:n},X_{j:n}}(x,y) = {} & \frac{n!}{(i-1)!(j-i-1)!(n-j)!} f(x)f(y) \\ & \times \{F(x)\}^{i-1}\{F(y)-F(x)\}^{j-i-1}\{1-F(y)\}^{n-j}, \\ & -\infty < x < y < \infty. \end{aligned} \tag{3.3}$$

3.1.2 The cdf of Order Statistics

The cdf of a single order statistic $X_{i:n}$ corresponding to the pdf in (3.2) could be obtained in the following manner. For any $1 \leq i \leq n$ and any $x \in \mathbb{R}$,

$$\begin{aligned} F_{X_{i:n}}(x) &= P\left\{X_{i:n} \leq x\right\} \\ &= P\left\{\text{at least } i \text{ of } X_1, \ldots, X_n \leq x\right\} \\ &= \sum_{r=i}^{n} P\left\{\text{exactly } r \text{ of } X_1, \ldots, X_n \leq x\right\} \\ &= \sum_{r=i}^{n} \frac{n!}{r!(n-r)!} \{F(x)\}^r \{1 - F(x)\}^{n-r}. \end{aligned}$$

The joint cdf of $X_{i:n}$ and $X_{j:n}$ corresponding to the pdf in (3.3) could be obtained by a direct argument (David and Nagaraja, 2003). For any $1 \leq i < j \leq n$ and any $x, y \in \mathbb{R}(x < y)$,

$$\begin{aligned} F_{X_{i:n},X_{j:n}}(x,y) &= P\left\{\text{at least } i \text{ of } X_1, \ldots, X_n \leq x, \text{at least } j \text{ of } X_1, \ldots, X_n \leq y\right\} \\ &= \sum_{s=j}^{n} \sum_{r=i}^{j} P\left\{\text{exactly } r \text{ of } X_1, \ldots, X_n \leq x, \text{exactly } s \text{ of } X_1, \ldots, X_n \leq y\right\} \\ &= \sum_{s=j}^{n} \sum_{r=i}^{j} \frac{n!}{r!(s-r)!(n-s)!} \{F(x)\}^r \{F(y) - F(x)\}^{s-r} \{1 - F(y)\}^{n-s}. \end{aligned}$$

3.1.3 The Markov Property of Order Statistics

The order statistics in a sample from an absolutely continuous population form a Markov chain.

Consider the conditional density of $X_{j:n}$, given $(X_{1:n} = x_1, \ldots, X_{i:n} = x_i)$, given by

$$
\begin{aligned}
& f_{X_{j:n}|X_{1:n}=x_1,\ldots,X_{i:n}=x_i}(x_j|x_1,\ldots,x_i) \\
=& \frac{f_{X_{1:n},\ldots,X_{i:n},X_{j:n}}(x_1,\ldots,x_i,x_j)}{f_{X_{1:n},\ldots,X_{i:n}}(x_1,\ldots,x_i)} \\
=& \frac{(n-i)!}{(j-i-1)!(n-j)!} \times \frac{f(x_j)\{F(x_j)-F(x_i)\}^{j-i-1}\{1-F(x_j)\}^{n-j}}{\{1-F(x_i)\}^{n-i}}, \\
& \qquad\qquad\qquad\qquad x_i < x_j, 1 \le i < j \le n. \qquad (3.4)
\end{aligned}
$$

Since Eq. (3.4) does not depend on $x_1, \ldots, x_{i-1}$ but depends only on x_i, we conclude the Markov dependence property of order statistics.

3.2 Moving Order Statistics

Order statistics or functions of order statistics in overlapping samples arise naturally in a number of contexts, with one principal area of application being to moving samples. “Moving samples have a long history in quality control and time series analysis” (David and Nagaraja, 2003). For example, the moving average control chart is used to assess the stability of a process. Order statistics in moving samples can be applied to indicate location and dispersion changes in a time series (Cleveland and Kleiner, 1975). “Moving order statistics are of interest primarily in graphical displays”(David and Rogers, 1983).

Let X_i $(i = 1, 2, ..., m)$ be a sequence of independent random variables with common pdf $f(x)$ and cdf $F(x)$. Let $S_n^{(l)}$=$(X_l, ..., X_{l+n-1})(l = 1, 2, ..., m-n+1)$ be the moving samples. Let $X_{r:n}^l$ denote the rth order statistic in $S_n^{(l)}$. Then, the first moving sample is $S_n^{(1)}$=$(X_1, ..., X_n)$ and the last moving sample is $S_n^{(m-n+1)}$=$(X_{m-n+1}, ..., X_m)$.

Suppose there is no overlapping between the first and last moving samples. Then, under H_0, they are identically distributed. Moreover, their order statistics are also identically distributed since they have the same sample size n. One may refer to Inagaki (1980), David and Rogers (1983) and David and Nagaraja (2003) for discussions on the distributions of order statistics in overlapping samples.

3.3 Proposed Test Statistic

The linear function of moving order statistics of the last moving sample

$$I = \sum_{j=1}^{n} c_j X_{j:n}^{m-n+1} = \sum_{j=1}^{n} c_j Y_j \quad (j = 1, 2, ..., n), \quad c_j \in \mathbb{R},$$

is a potential test statistic to test the trend in a sequence. Here, let $Y_j = X_{j:n}^{m-n+1}$ $(j = 1, 2, ..., n)$, just for convenience. Then, under H_0, $\sum_{j=1}^{n} c_j X_{j:n}^{1}$ and $\sum_{j=1}^{n} c_j X_{j:n}^{m-n+1}$ are identically distributed if there is no overlapping between them.

3.4 Exact Null distribution

In this section, we introduce L-statistics and difference matrix and then derive the exact distribution of the proposed test statistic.

3.4.1 L-statistics

Let $\boldsymbol{Y} = (Y_1, ..., Y_n)'$ be the vector of order statistics corresponding to the data $\boldsymbol{X} = (X_1, ..., X_n)'$, and let $\boldsymbol{I} = \Omega \boldsymbol{Y}$ be a generic vector of L-statistics. The linear operator Ω maps $\boldsymbol{Y}$ onto L and is called the weight matrix of L.

3.4.2 Difference Matrix

The matrix $\Delta = \{d_{ij}\}$ is said to be a difference matrix of size $(n-1) \times n$ if $d_{ii} = -1, d_{ii+1} = 1$ for $i = 1, ..., n-1$ and 0 elsewhere.

3.4.3 The Distribution of $\boldsymbol{I}$

First, we show that $\boldsymbol{I} \stackrel{d}{=} [\Omega \boldsymbol{X} \mid \Delta \boldsymbol{X} \geq 0]$. Let $\boldsymbol{X} = (X_1, ..., X_n)'$ be a random vector whose components are i.i.d., and let $\boldsymbol{I} = (L_1, ..., L_p)'$ be the corresponding vector of L-statistics with weight matrix Ω. Then,

$$\boldsymbol{I} \stackrel{d}{=} [\Omega \boldsymbol{X} \mid \Delta \boldsymbol{X} \geq \mathbf{0}]. \tag{3.5}$$

Proof Crocetta and Loperfido (2005) obtained the distribution of $\boldsymbol{I}$ in the proof of their Theorem 1.

First, consider the $n!$ permutations of $X_1, ..., X_n$, assign a progressive number to each permutation, and denote by $\boldsymbol{X_i} = (X_{i:1}, ..., X_{i:n})$ the i-th permutation of the elements of $\boldsymbol{X} = (X_1, ..., X_n)$. Apply now the theorem of total probabilities to get

$$f_{\boldsymbol{I}}(\boldsymbol{a}) = \sum_{i=1}^{n!} f_{\Omega \boldsymbol{X_i}}(\boldsymbol{a} \mid X_{i:1} \leq ... \leq X_{i:n}) P(X_{i:1} \leq ... \leq X_{i:n}).$$

By assumption, $X_1, ..., X_n$ are independent and identically distributed. So, it follows that

$$f_{\Omega \boldsymbol{X_i}}(\boldsymbol{a} \mid X_{i:1} \leq ... \leq X_{i:n}) = f_{\Omega \boldsymbol{X}}(\boldsymbol{a} \mid X_1 \leq ... \leq X_n),$$

$$P(X_{i:1} \leq ... \leq X_{i:n}) = \frac{1}{n!},$$

and consequently

$$f_{\boldsymbol{I}}(\boldsymbol{a}) = \frac{1}{n!}\sum_{i=1}^{n!} f_{\Omega \boldsymbol{X}_i}(\boldsymbol{a} \mid X_{i:1} \leq ... \leq X_{i:n}) = f_{\Omega \boldsymbol{X}}(\boldsymbol{a} \mid X_1 \leq ... \leq X_n).$$

Inequalities $X_1 \leq ... \leq X_n$ and $X_2 - X_1 \geq 0, ..., X_n - X_{n-1} \geq 0$ are equivalent, and therefore

$$f_{\boldsymbol{I}}(\boldsymbol{a}) = f_{\Omega \boldsymbol{X}}(\boldsymbol{a} \mid X_2 - X_1 \geq 0, ..., X_n - X_{n-1} \geq 0).$$

By definition, Δ is the difference matrix of size $n-1 \times n$. Then,

$$\Delta \boldsymbol{X} = \begin{pmatrix} -1 & 1 & 0 & \dots & \dots \\ 0 & -1 & 1 & \dots & \dots \\ 0 & 0 & -1 & 1 & \dots \\ \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & 0 & -1 & 1 \end{pmatrix} \begin{pmatrix} X_1 \\ X_2 \\ X_3 \\ \dots \\ X_n \end{pmatrix} = \begin{pmatrix} X_2 - X_1 \\ X_3 - X_2 \\ X_4 - X_3 \\ \dots \\ X_n - X_{n-1} \end{pmatrix}.$$

The inequality $X_2 - X_1 \geq 0, ..., X_n - X_{n-1} \geq 0$ and $\Delta \boldsymbol{X} \geq \boldsymbol{0}$ are equivalent. Thus,

$$f_{\boldsymbol{I}}(\boldsymbol{a}) = f_{\Omega \boldsymbol{X}}(\boldsymbol{a} \mid \Delta \boldsymbol{X} \geq \boldsymbol{0})$$

so that

$$\boldsymbol{I} \stackrel{d}{=} [\Omega \boldsymbol{X} \mid \Delta \boldsymbol{X} \geq \boldsymbol{0}],$$

which completes the proof.

Now we derive the exact distribution of $\boldsymbol{I}$. Let $\boldsymbol{X} = (X_1, ..., X_n)'$ be a random sample from a standard normal distribution and let $\boldsymbol{I} = (L_1, ..., L_p)'$ be the corresponding vector of L-statistics with weight matrix $\Omega \in \mathbb{R}^{p \times n}$. Then, the pdf of $\boldsymbol{I}$

is

$$f_{\boldsymbol{I}}(\boldsymbol{a}) = n!\phi_p(\boldsymbol{a};\Omega\Omega')\Phi_{n-1}\left[\Delta\Omega'(\Omega\Omega')^{-1}\boldsymbol{a};\Delta\Delta' - \Delta\Omega'(\Omega\Omega')^{-1}\Omega\Delta'\right],$$

i.e.,

$$\boldsymbol{I} \sim CSN_{p,n-1}(\mathbf{0},\Omega\Omega',\Delta\Omega'(\Omega\Omega')^{-1},\mathbf{0},\Delta\Delta' - \Delta\Omega'(\Omega\Omega')^{-1}\Omega\Delta'), \tag{3.6}$$

where Δ is the difference matrix of dimension $(n-1)\times n$.

Proof Given that $\Delta\boldsymbol{X} \sim N_{n-1}(\mathbf{0},\Delta\Delta')$ and $\Omega\boldsymbol{X} \sim N_p(\mathbf{0},\Omega\Omega')$, the joint distribution of $\Delta\boldsymbol{X}$ and $\Omega\boldsymbol{X}$ is

$$\begin{pmatrix} \Delta\boldsymbol{X} \\ \Omega\boldsymbol{X} \end{pmatrix} \sim N_{(n-1)+p}\left[\begin{pmatrix} \mathbf{0} \\ \mathbf{0} \end{pmatrix},\begin{pmatrix} \Delta\Delta' & \Delta\Omega' \\ \Omega\Delta' & \Omega\Omega' \end{pmatrix}\right]. \tag{3.7}$$

By (2.4), the conditional distribution of $\Delta\boldsymbol{X}$, given $\Omega\boldsymbol{X} = \boldsymbol{a}$, is then

$$\Delta\boldsymbol{X}|\Omega\boldsymbol{X} = \boldsymbol{a} \sim N_{n-1}(\Delta\Omega'(\Omega\Omega')^{-1}\boldsymbol{a},\Delta\Delta' - \Delta\Omega'(\Omega\Omega')^{-1}\Omega\Delta').$$

Thus, the conditional density of $\Omega\boldsymbol{X}$, given $\Delta\boldsymbol{X} \geq 0$, is obtained as

$$\begin{aligned}
&f_{\Omega\boldsymbol{X}|\Delta\boldsymbol{X}\geq\mathbf{0}}(\boldsymbol{a}|\Delta\boldsymbol{X} \geq 0) \\
&= \frac{f_{\Omega\boldsymbol{X}}(\boldsymbol{a})}{P(\Delta\boldsymbol{X} \geq 0)}P(\Delta\boldsymbol{X} \geq 0|\Omega\boldsymbol{X} = \boldsymbol{a}) \\
&= \frac{f_{\Omega\boldsymbol{X}}(\boldsymbol{a})}{P(\Delta\boldsymbol{X} \geq 0)}P(-\Delta\boldsymbol{X} < 0|\Omega\boldsymbol{X} = \boldsymbol{a}) \\
&= K\phi_p(\boldsymbol{a};\mathbf{0},\Omega\Omega')\Phi_{n-1}(\mathbf{0};-\Delta\Omega'(\Omega\Omega')^{-1}\boldsymbol{a},\Delta\Delta' - \Delta\Omega'(\Omega\Omega')^{-1}\Omega\Delta') \\
&= K\phi_p(\boldsymbol{a};\mathbf{0},\Omega\Omega')\Phi_{n-1}(\Delta\Omega'(\Omega\Omega')^{-1}\boldsymbol{a};\mathbf{0},\Delta\Delta' - \Delta\Omega'(\Omega\Omega')^{-1}\Omega\Delta'),
\end{aligned}$$

where

$$K = \frac{1}{P(\Delta \boldsymbol{X} \geq \mathbf{0})} = \frac{1}{n!} = \frac{1}{\Phi_p(\mathbf{0}; \mathbf{0}, \Delta\Delta')}.$$

Thus,

$$\boldsymbol{I} \sim CSN_{p,n-1}(\mathbf{0}, \Omega\Omega', \Delta\Omega'(\Omega\Omega')^{-1}, \mathbf{0}, \Delta\Delta' - \Delta\Omega'(\Omega\Omega')^{-1}\Omega\Delta').$$

Finally, we find the exact distribution of the proposed test statistic. Recall that the proposed test statistic

$$I = \sum_{j=1}^{n} c_j X_{j:n}^{m-n+1} = \sum_{j=1}^{n} c_j Y_j \quad (j = 1, 2, ..., n), \quad c_j \in \mathbb{R},$$

is a linear function of moving order statistics. Its distribution, under the null hypothesis, is the case of Eq. (3.6) with $p = 1$ and $\Omega = \boldsymbol{c}' = (c_1, c_2, \ldots, c_n)$. Then,

$$f_I(a) = n!\phi(a; \boldsymbol{c}'\boldsymbol{c})\Phi_{n-1}\left[\Delta\boldsymbol{c}(\boldsymbol{c}'\boldsymbol{c})^{-1}a; \Delta\Delta' - \Delta\boldsymbol{c}(\boldsymbol{c}'\boldsymbol{c})^{-1}\boldsymbol{c}'\Delta'\right],$$

i.e.,

$$I \sim CSN_{1,n-1}(0, \boldsymbol{c}'\boldsymbol{c}, \Delta\boldsymbol{c}(\boldsymbol{c}'\boldsymbol{c})^{-1}, \mathbf{0}, \Delta\Delta' - \Delta\boldsymbol{c}(\boldsymbol{c}'\boldsymbol{c})^{-1}\boldsymbol{c}'\Delta'), \tag{3.8}$$

where Δ is the difference matrix of dimension $(n-1) \times n$. If $c_1 = c_2 = \ldots = c_n = 1$, then $\Delta\boldsymbol{c} = \mathbf{0}$ and $\boldsymbol{c}'\boldsymbol{c} = n$, and in this case

$$f_I(a) = n!\phi(a; n)\Phi_{n-1}\left[\mathbf{0}; \Delta\Delta'\right] = n!\phi(a; n)\frac{1}{n!} = \phi(a; n),$$

i.e.,

$$I \sim CSN_{1,n-1}(0, n, \mathbf{0}, \mathbf{0}, \Delta\Delta') \tag{3.9}$$

or simply $N(0, n)$.

3.5 Proposed Test Procedure

Recall the hypotheses testing problem we are interested in is

$$H_0 : \beta = 0 \quad \text{vs.} \quad H_1 : \beta > 0.$$

The null hypothesis means no trend and the alternative hypothesis represents that there is an increasing trend.

Under H_0, $\sum_{j=1}^{n} c_j X_{j:n}^{1}$ and $\sum_{j=1}^{n} c_j X_{j:n}^{m-n+1}$ are identically distributed if there is no overlapping between the first and last moving samples. Let $I = \sum_{j=1}^{n} c_j X_{j:n}^{m-n+1}$. Then, by (3.8), the test statistic $I \sim CSN_{1,n-1}(0, \boldsymbol{c}'\boldsymbol{c}, \Delta\boldsymbol{c}(\boldsymbol{c}'\boldsymbol{c})^{-1}, \mathbf{0}, \Delta\Delta' - \Delta\boldsymbol{c}(\boldsymbol{c}'\boldsymbol{c})^{-1}\boldsymbol{c}'\Delta')$.

Evidently, large values of I lead to the rejection of H_0 and in favor of H_1. For specified values of m, n, the weights $\boldsymbol{c}' = (c_1, c_2, \ldots, c_n)$, and the level of significance α, the critical region will be of the form

$$\left\{ (x_{m-n+1}, x_{m-n+2}, \ldots, x_m) : \sum_{j=1}^{n} c_j x_{j:n}^{m-n+1} \geq s \right\},$$

where

$$\alpha = P(I \geq s | H_0 : \beta = 0) = P\left(\sum_{j=1}^{n} c_j X_{j:n}^{m-n+1} \geq s | H_0 : \beta = 0 \right).$$

We find the critical value s by Monte Carlo simulations since it is computationally hard to determine it directly from the density .

Chapter 4

Empirical Evaluation

4.1 Introduction

In this chapter we use Monte Carlo simulations to find the critical values of the CSN distribution and then evaluate the performance of the proposed test by means of a power study.

In order to determine the critical value s, we need to do simulations since it is computationally hard to find it directly from the density. We also need to carry out a simulation study to examine the power performance of the proposed test for various alternatives.

Consider the observations $z_1, ..., z_N$ as independent realizations of the CSN random variable. For any given significance level α, the critical value s is determined by the following equation

$$\alpha = P(\text{reject } H_0 | H_0 \text{ is true}) = P(I \geq s | H_0 : \beta = 0) = \frac{1}{N} \sum_{i=1}^{N} I_s \{z_i \geq s\},$$

where $I_s(\cdot)$ is an indicator function given by

$$I_s(z_i) = \begin{cases} 1 & z_i \geq s \\ 0 & z_i < s \end{cases}, \qquad i = 1, \ldots, N.$$

Then, consider the observations $x_1, ..., x_m$ as independent realizations of the time series model in (1.1):

$$X_i = \beta t(i) + \varepsilon_i, \qquad i = 1, 2, ..., m,$$

where $\varepsilon_i \sim i.i.d.\ N(0,1)$ and $\beta > 0$ which marks an increasing trend in location. The trend function $t(i)$ takes linear, logarithmic and quadratic forms, respectively. Let $S_n^{(l)} = (x_l, ..., x_{l+n-1})$ $(l = 1, 2, ..., m-n+1)$ be the l-th moving sample. Let $x_{r:n}^l$ denote the r-th order statistic in $S_n^{(l)}$. The last moving sample is $S_n^{(m-n+1)}$=$(x_{m-n+1}, ..., x_m)$. Then, the power of the test is

$$\pi = P(\text{reject } \ H_0|H_1 \ \text{ is true}) = P\left(\sum_{j=1}^{n} c_j x_{j:n}^{m-n+1} \geq s|H_1 : \beta = \beta_1\right),$$

where $\beta_1 > 0$.

4.2 Algorithms

In this section, two algorithms are presented for the purpose of generating random numbers from the CSN distribution.

4.2.1 The Decomposition of $\boldsymbol{I}$

Allard and Naveau (2007) introduced a decomposition of the CSN random vector, which forms the direct theoretical foundation for our simulation. Let us decompose $\boldsymbol{I}$ in (3.6).

Recall $\Delta\boldsymbol{X} \sim N_{n-1}(\mathbf{0}, \Delta\Delta')$ in Eq. (3.5). We consider the augmented Gaussian vector $((\Delta\boldsymbol{X})', U')'$ such that

$$\begin{pmatrix} \Delta\boldsymbol{X} \\ U \end{pmatrix} \sim N_{(n-1)+p}\left[\begin{pmatrix} \mathbf{0} \\ \mathbf{0} \end{pmatrix}, \begin{pmatrix} \Delta\Delta' & 0 \\ 0 & I_p \end{pmatrix}\right],$$

where $U \sim N_p(\mathbf{0}, \boldsymbol{I_p})$, $\Delta\boldsymbol{X}$ and U are independent.

Now, recall that $\boldsymbol{I} \stackrel{d}{=} [\Omega\boldsymbol{X} \mid \Delta\boldsymbol{X} \geq 0]$ in (3.6). Note that the Gaussian vector $\Omega\boldsymbol{X} \sim N_p(\mathbf{0}, \Omega\Omega')$ can be expressed as

$$\Omega\boldsymbol{X} = F\Delta\boldsymbol{X} + G^{1/2}U,$$

where $F = \Omega\Delta'(\Delta\Delta')^{-1}$, $G = \Omega\Omega' - \Omega\Delta'(\Delta\Delta')^{-1}\Delta\Omega'$, $G^{1/2}$ is lower-triangular and the Cholesky factorization of G, a symmetric and positive definite matrix, and $G = G^{1/2}(G^{1/2})'$.

It is easy to verify that $\Omega\boldsymbol{X} \stackrel{d}{=} F\Delta\boldsymbol{X} + G^{1/2}U$, since $E(F\Delta\boldsymbol{X} + G^{1/2}U) =$

$F\Delta E(\boldsymbol{X}) + G^{1/2}E(U) = 0$ and

$$\begin{aligned}
& Var(F\Delta\boldsymbol{X} + G^{1/2}U) \\
&= Var(F\Delta\boldsymbol{X}) + Var(G^{1/2}U) \\
&= F\Delta Var(\boldsymbol{X})(F\Delta)' + G^{1/2}Var(U)(G^{1/2})' \\
&= \Omega\Delta'(\Delta\Delta')^{-1}\Delta I(\Omega\Delta'(\Delta\Delta')^{-1}\Delta)' + G^{1/2}I(G^{1/2})' \\
&= \Omega\Delta'(\Delta\Delta')^{-1}\Delta\Omega' + \Omega\Omega' - \Omega\Delta'(\Delta\Delta')^{-1}\Delta\Omega' \\
&= \Omega\Omega'.
\end{aligned}$$

Therefore, it follows that

$$\boldsymbol{I} = F[\Delta\boldsymbol{X} \mid \Delta\boldsymbol{X} \geq 0] + G^{1/2}U. \tag{4.1}$$

4.2.2 Algorithm 1

This is an algorithm used by Iversen (2010), which he wrote in matlab. We generate the CSN random variable in (3.6). Let us denote (3.7) by

$$\begin{pmatrix} \boldsymbol{v} \\ \boldsymbol{t} \end{pmatrix} \sim N_{(n-1)+p}\left[\begin{pmatrix} \mathbf{0} \\ \mathbf{0} \end{pmatrix}, \begin{pmatrix} \Sigma_v & \Gamma_{vt} \\ \Gamma_{tv} & \Sigma_t \end{pmatrix}\right],$$

where $\boldsymbol{v} \triangleq \Delta\boldsymbol{X}$, $\boldsymbol{t} \triangleq \Omega\boldsymbol{X}$, $\Sigma_v \triangleq \Delta\Delta'$, $\Gamma_{vt} \triangleq \Delta\Omega'$, $\Gamma_{tv} \triangleq \Omega\Delta'$ and $\Gamma_t \triangleq \Omega\Omega'$. These notations are used in the description of the algorithm as well as in our R program.

Step 1 Generate $n-1$ independent $N(0,1)$ observations to get a vector $\boldsymbol{z_1}$ of dimension $n-1$;

Step 2 Set $\boldsymbol{v} = A_1\boldsymbol{z_1}$, where A_1 is lower-triangular and the Cholesky factorization of $\Delta\Delta'$ such that $A_1A_1' = \Delta\Delta' \triangleq \Sigma_v$;

Step 3 If any element $v_i < 0$ ($v_i \in \boldsymbol{v}, i = 1, \dots, n-1$), repeat Steps 1 and 2 until $\boldsymbol{v} \geq 0$;

Step 4 Generate p $N(0,1)$ observations to get a vector $\boldsymbol{z_2}$ of dimension p;

Step 5 By (4.1), we set $\boldsymbol{I} = F\boldsymbol{v} + A_2\boldsymbol{z_2}$. We have $\Delta\Omega' \triangleq \Gamma_{vt}$, and then $F = {\Gamma_{vt}}'{\Sigma_v}^{-1} = \Omega\Delta'(\Delta\Delta')^{-1}$. A_2 is lower-triangular, which is the Cholesky factorization of $G = \Sigma_t - {\Gamma_{vt}}'{\Sigma_v}^{-1}\Gamma_{vt} = \Omega\Omega' - \Omega\Delta'(\Delta\Delta')^{-1}\Delta\Omega'$ such that $A_2A_2' = G$.

Then, $\boldsymbol{I}$ is the required sample from the $CSN_{p,n-1}(\mathbf{0}, \Omega\Omega', \Delta\Omega'(\Omega\Omega')^{-1}, \mathbf{0}, \Delta\Delta' - \Delta\Omega'(\Omega\Omega')^{-1}\Omega\Delta')$ distribution.

4.2.3 Algorithm 2

The disadvantage of Algorithm 1 is that it requires the element $v_i \geq 0$ for all i simultaneously, which involves Steps 1 to 3. This process, generating a random vector from a normal distribution till all its elements are bigger than 0, is time-consuming, and it becomes almost impossible when $\boldsymbol{v}$ is high-dimensional. So as n increases, the algorithm becomes very slow. In order to overcome this drawback, in Algorithm 2, $\boldsymbol{v}$ is generated from a multivariate left truncated normal distribution truncated at $\mathbf{0}$. This algorithm turns out to be quite efficient.

So, we just replace Steps 1 to 3 in Algorithm 1 by the following Step 1:

Step 1 Generate 1 vector $\boldsymbol{v}$ from a multivariate left truncated normal distribution based on $\boldsymbol{v} \sim N_{n-1}(\mathbf{0}, \Delta\Delta')$ truncated at $\mathbf{0}$ to get $\boldsymbol{v} \geq 0$ (we use R command "tmvtnorm");

Step 2 same as Step 4 in Algorithm 1;

Step 3 same as Step 5 in Algorithm 1.

4.2.4 Comparison

1. Time Efficiency Comparison

To compare the time efficiency of the two algorithms, we carried out 1000 simulations. In our setup, $p = 1$, $\Omega = \boldsymbol{c}' = (1, 2, \ldots, n)$, and by (3.8) $I \sim CSN_{1,n-1}(0, \boldsymbol{c}'\boldsymbol{c}, \Delta\boldsymbol{c}(\boldsymbol{c}'\boldsymbol{c})^{-1}, \mathbf{0}, \Delta\Delta' - \Delta\boldsymbol{c}(\boldsymbol{c}'\boldsymbol{c})^{-1}\boldsymbol{c}'\Delta')$.

As seen in Table 4.1, for $n = 2(1)4$, the two algorithms are very close. Algorithm 1 is slightly faster than Algorithm 2. For $n \geq 5$, Algorithm 2 becomes more and more efficient than Algorithm 1. For $n = 9$, the running time of Algorithm 1 is about 1 hour, while that of Algorithm 2 is just 1 second. From Figure 4.1, we can see the difference in runtime increases dramatically since the running time of Algorithm 1 grows exponentially. The simulations are performed in the R statistical computing environment. A computer with Intel Core i5-2430 2.4GHz Dual-Core Processor and 8GB DDR3 RAM was used.

Table 4.1: Runtime in seconds (s) for the algorithms in 1000 simulations

n	Algorithm 1	Algorithm 2	Difference
2	0.04	0.45	0.41
3	0.08	0.61	0.53
4	0.25	0.64	0.39
5	1.20	0.71	-0.49
6	6.85	0.80	-6.05
7	48.54	0.86	-47.68
8	397.20	0.92	-396.28
9	3556.71	1.00	-3555.71

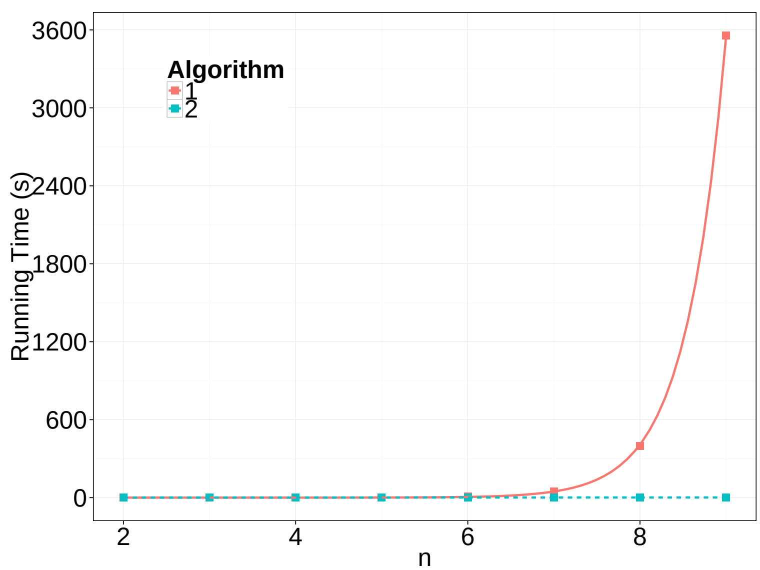

Figure 4.1: Runtime in seconds (s) for Algorithm 1 (represented by the solid red line) and Algorithm 2 (represented by the dashed blue line) in 1000 simulations for $n = 2(1)9$

2. Distribution Comparison

For a specific sample size n, the realizations generated by the two algorithms indeed come from the same distribution. The plots of overlapping density curves of I for $n = 2(1)9$ are provided in Figures 4.2 and 4.3. In each plot, we observe that the two density curves are quite close. The corresponding results of Kolmogorov-Smirnov tests are given in Table 4.2. This test does not reject that the realizations generated by the two algorithms come from the same distribution as the distance between the empirical c.d.f's is insignificantly small. Indeed, the p-values in the last column show no evidence against the null hypothesis.

Table 4.2: Results of the Kolmogorov-Smirnov test for $n = 2(1)9$

	Kolmogorov-Smirnov test	
n	test statistic values	p-values
2	0.031	0.7226
3	0.027	0.8593
4	0.024	0.9356
5	0.035	0.5727
6	0.022	0.9689
7	0.037	0.5004
8	0.038	0.4658
9	0.024	0.9356

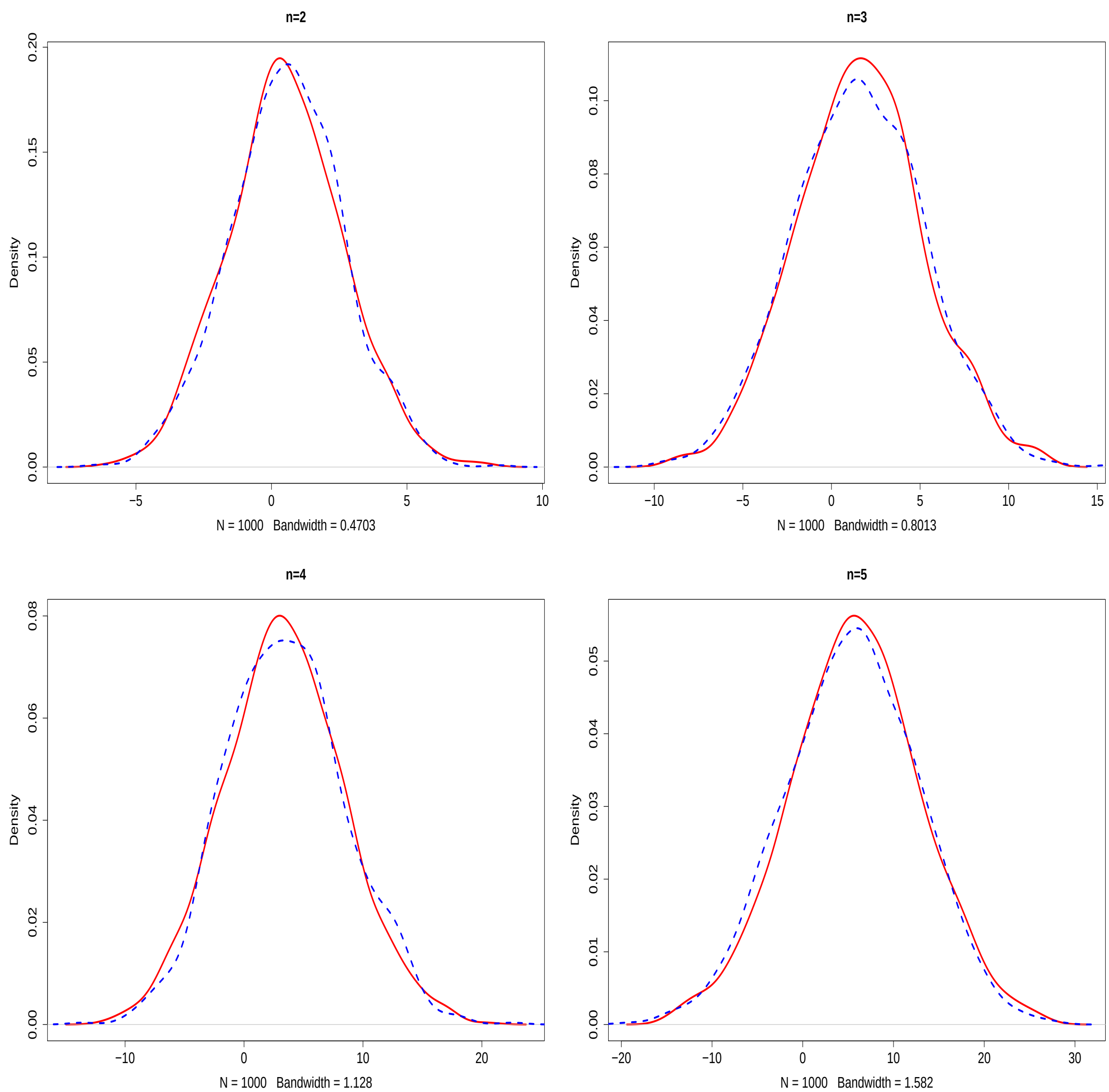

Figure 4.2: Overlapping density curves of the CSN distribution generated by Algorithm 1 (represented by the solid red line) and Algorithm 2 (represented by the dashed blue line) in 1000 simulations for $n = 2, 3, 4, 5$

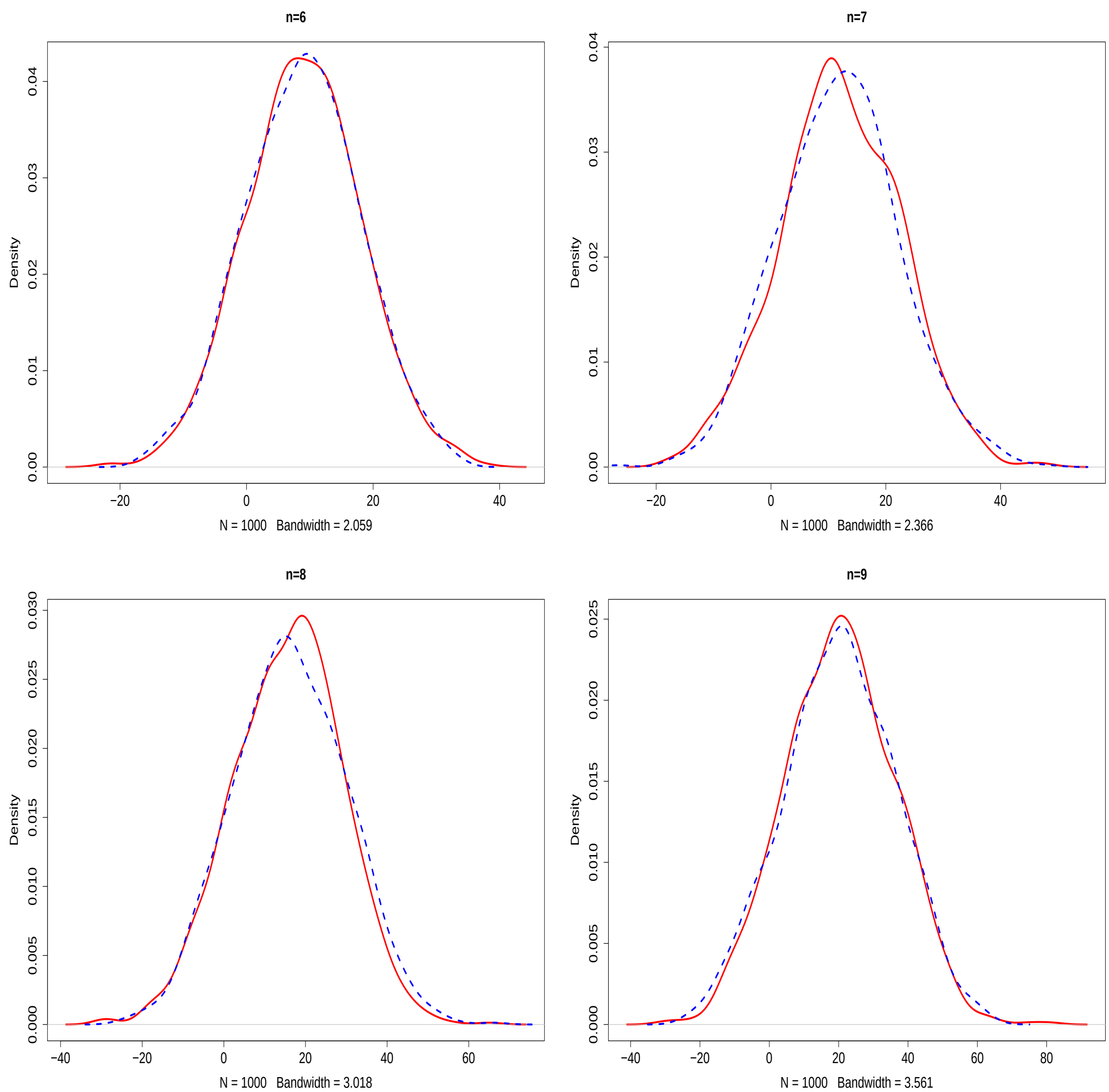

Figure 4.3: Overlapping density curves of the CSN distribution generated by Algorithm 1 (represented by the solid red line) and Algorithm 2 (represented by the dashed blue line) in 1000 simulations for $n = 6, 7, 8, 9$

4.3 Empirical Type I Error Rate

Before conducting the power study, we need to evaluate the empirical level with the nominal α set as 0.05. Empirical Type I error rate can be calculated as the proportion of significant test statistics under the null hypothesis of no trend among the replicates.

Table 4.3: Average Empirical Type I Error Rate

m	Weight	Average Empirical α
10	c(1:n)	0.050048
10	rep(1,n)	0.049947
100	c(1:n)	0.050142
100	rep(1,n)	0.050305

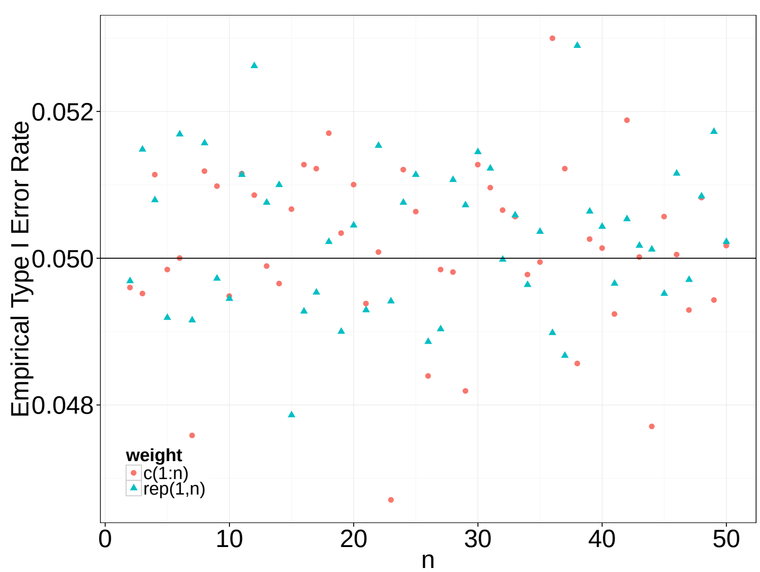

Figure 4.4: Empirical Type I Error Rate for m=100 with weights c(1:n) (represented by the round red points) and weights rep(1,n) (represented by the triangle blue points) in 58,000 simulations

The average empirical Type I error results for $m = 10$ of 1,000,000 simulations and for $m = 100$ of 58,000 simulations are given in Table 4.3. Although three of them are slightly larger than 0.05, they are not statistically significant. The average empirical α for $m = 10$ with weights rep(1,n) is below 0.05. Empirical Type I Error Rates for m=100 with both type of weights are plotted in Figure 4.4. We can see that they are randomly distributed around 0.05.

4.4 Power Study

This study aims to evaluate the proposed test by examining the rejection rates under $H_1 : \beta > 0$.

4.4.1 Setup

To assess the performance of the proposed test and compare it with those of the tests presented earlier in Hofmann and Balakrishnan (2006), we use the similar simulation setup.

An initial simulation under $H_0 : \beta = 0$ determines the critical values at the $\alpha = 0.05$ level of significance. Two types of weights, $\boldsymbol{c}' = (1, 2, \ldots, n)$ denoted by c(1:n) and $\boldsymbol{c}' = (1, 1, \ldots, 1)$ denoted by rep(1,n), are used in construction of the test statistic and the corresponding critical values are determined. We then simulate the power under the alternative hypotheses with different trend functions. For the purpose of comparison, the trend coefficient β is so chosen that the power of Kendall's Q falls between 0.5 and 0.6.

For $m = 10$ and $n = 2(1)5$, we carry out 1,000,000 simulations by Algorithm 1 and Algorithm 2, respectively. To find the power values, we generate independent realizations from the following time series models:

$$\begin{aligned}
&\text{Linear trend:} && X_i = 0.25i + \varepsilon_i, \\
&\text{Log trend:} && X_i = 1.2log(i) + \varepsilon_i, \\
&\text{Quadratic trend:} && X_i = 0.023(i^2 + 0.023i) + \varepsilon_i,
\end{aligned}$$

where $\varepsilon_i \sim i.i.d.\ N(0,1)$, $i = 1, 2, ..., 10$.

For $m = 100$ and $n = 2(1)50$, we performed 58,000 simulations by Algorithm 2 (due to the time inefficiency of Algorithm 1, it was not used when n was large). To calculate the power values, we generated independent realizations from the following time series models:

$$\begin{aligned}
&\text{Linear trend:} && X_i = 0.006i + \varepsilon_i, \\
&\text{Log trend:} && X_i = 0.23log(i) + \varepsilon_i, \\
&\text{Quadratic trend:} && X_i = 0.00006(i^2 + 0.00006i) + \varepsilon_i,
\end{aligned}$$

where $\varepsilon_i \sim i.i.d.\ N(0,1)$, $i = 1, 2, ..., 100$.

4.4.2 Results and Comments

For $\alpha = 0.05$, $m = 10$, and $n = 2(1)5$, the results of 1,000,000 simulations are given in Table 4.4 . In this scenario, the test can detect the trends with large power values. As n increases, the power increases as well. For example, in the column of linear trend of Algorithm 1, the power with weights rep(1,2) is 0.958 while the power with rep(1,5) is 0.998. The power values with weights rep(1,n) are slightly larger than those with weights c(1:n). For example, in the column of quadratic trend of

Algorithm 2, the power with weights rep(1,2) is 0.905 while that with weight c(1:2) is 0.896. There's almost no difference between Algorithms 1 and 2 in terms of power values.

Table 4.4: Power of the proposed test in $1,000,000$ simulations when $\alpha = 0.05$ and $m = 10$

Algorithm 1					
			Power		
n	Weights	Critical value	Linear	Log	Quadratic
2	c(1:2)	4.124	0.951	0.982	0.896
	rep(1,2)	2.324	0.958	0.985	0.904
3	c(1:3)	7.577	0.984	0.997	0.939
	rep(1,3)	2.848	0.988	0.998	0.947
4	c(1:4)	11.972	0.993	0.999	0.954
	rep(1,4)	3.296	0.995	1.000	0.960
5	c(1:5)	17.235	0.996	1.000	0.957
	rep(1,5)	3.686	0.998	1.000	0.960
Algorithm 2					
			Power		
n	Weights	Critical value	Linear	Log	Quadratic
2	c(1:2)	4.125	0.951	0.982	0.896
	rep(1,2)	2.323	0.957	0.985	0.905
3	c(1:3)	7.599	0.984	0.997	0.939
	rep(1,3)	2.849	0.988	0.998	0.947
4	c(1:4)	11.952	0.993	1.000	0.954
	rep(1,4)	3.293	0.995	1.000	0.960
5	c(1:5)	17.219	0.996	1.000	0.957
	rep(1,5)	3.677	0.998	1.000	0.961

For $\alpha = 0.05$, $m = 100$ and $n = 2(1)50$, the results of 58,000 simulations are showed in Figures 4.5 and 4.6 and in Table 4.5. The power increases is seen to increase with n. In Figure 4.5, the power curve with weights rep(1,n) is above that with weights c(1:n). As seen in Table 4.5, the power values with weights rep(1,n) are slightly larger than those with weights c(1:n). For instance, in the column of linear trend, the power with weights rep(1,25) is 0.845 while that with weight c(1:25) is 0.796. Figure 4.6 shows, for both sets of weights, the proposed test performs consistently best under log trend and worst under quadratic trend.

Compared with the nonparametric test statistics in Hofmann and Balakrishnan (2006) (see Appendix B TableB.1), which require the whole data set, the proposed test achieves much higher power by using only one-quarter of the full data set under the normal distribution assumption.

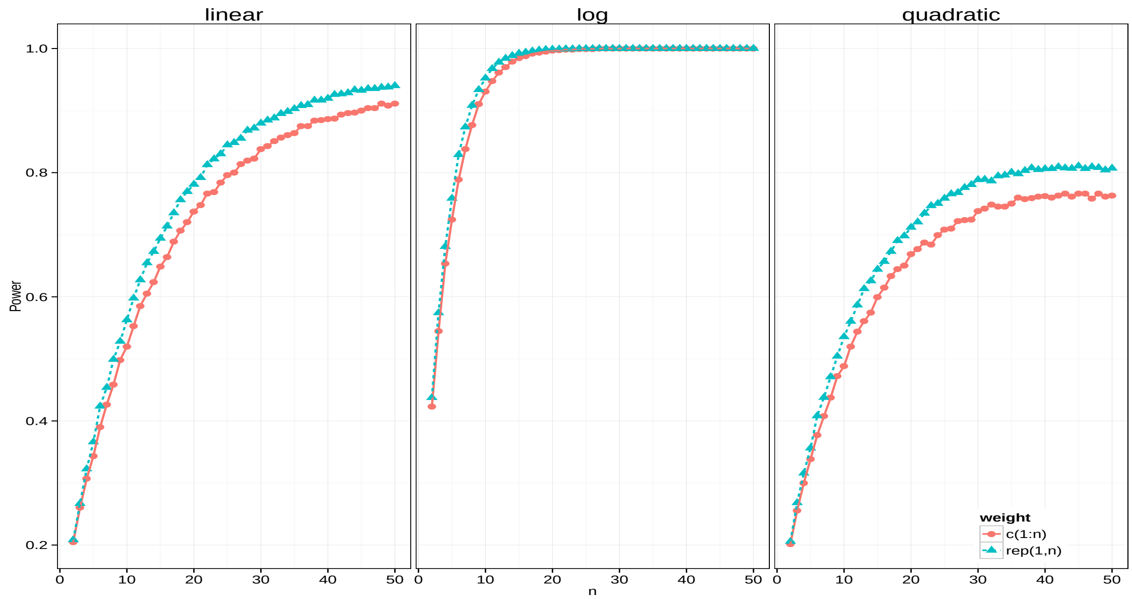

Figure 4.5: Power of the proposed test in 58,000 simulations by trends when $\alpha = 0.05$ and $m = 100$

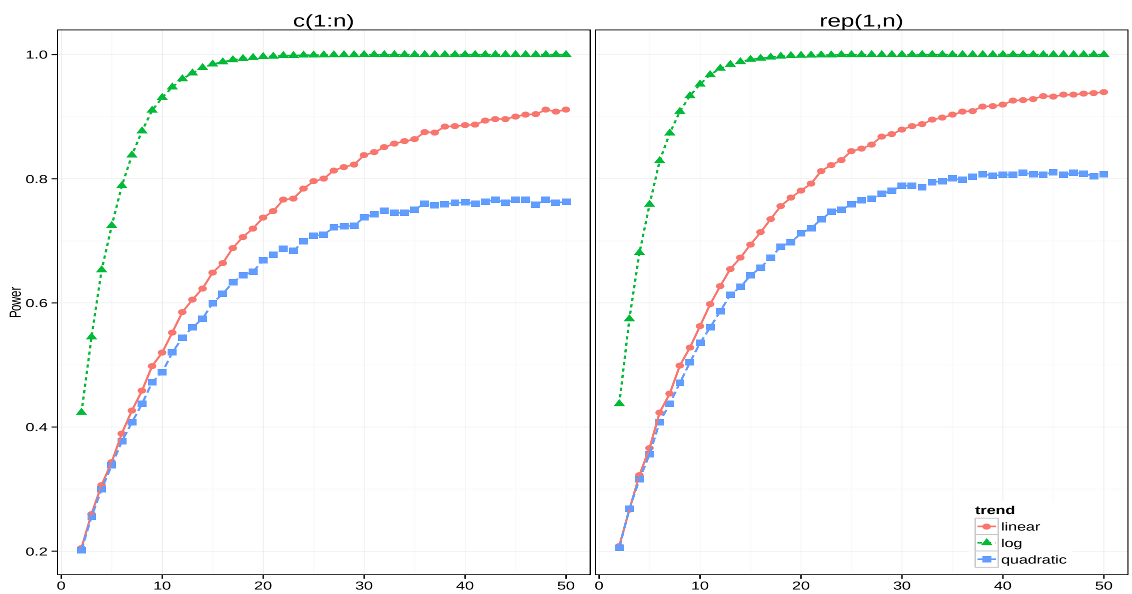

Figure 4.6: Power of the proposed test in 58,000 simulations by weights when $\alpha = 0.05$ and $m = 100$

Table 4.5: Power of the proposed test in 58,000 simulations when $\alpha = 0.05$ and $m = 100$

Algorithm 2					
			Power		
n	Weights	Critical value	Linear	Log	Quadratic
2	c(1:2)	4.129	0.205	0.423	0.202
	rep(1,2)	2.338	0.208	0.437	0.206
3	c(1:3)	7.582	0.260	0.545	0.256
	rep(1,3)	2.844	0.267	0.574	0.268
4	c(1:4)	11.921	0.307	0.653	0.300
	rep(1,4)	3.282	0.323	0.681	0.316
5	c(1:5)	17.238	0.344	0.724	0.338
	rep(1,5)	3.699	0.366	0.759	0.356
6	c(1:6)	23.306	0.390	0.789	0.377
	rep(1,6)	3.982	0.424	0.829	0.408
7	c(1:7)	30.233	0.427	0.838	0.408
	rep(1,7)	4.361	0.454	0.873	0.438
8	c(1:8)	37.949	0.459	0.877	0.437
	rep(1,8)	4.644	0.499	0.908	0.471
9	c(1:9)	46.243	0.498	0.910	0.472
	rep(1,9)	4.965	0.528	0.934	0.504
10	c(1:10)	56.038	0.520	0.931	0.488
	rep(1,10)	5.213	0.563	0.952	0.536
11	c(1:11)	65.709	0.552	0.948	0.520
	rep(1,11)	5.450	0.598	0.968	0.560
12	c(1:12)	76.426	0.585	0.961	0.544
	rep(1,12)	5.671	0.627	0.978	0.587
13	c(1:13)	88.160	0.605	0.970	0.560
	rep(1,13)	5.886	0.655	0.984	0.613
14	c(1:14)	100.838	0.623	0.979	0.575
	rep(1,14)	6.184	0.673	0.988	0.626
15	c(1:15)	113.362	0.649	0.985	0.600
	rep(1,15)	6.391	0.694	0.992	0.644
16	c(1:16)	127.418	0.664	0.988	0.615
	rep(1,16)	6.613	0.714	0.994	0.657
17	c(1:17)	141.517	0.689	0.992	0.633

Continued

Algorithm 2					
			Power		
n	Weights	Critical value	Linear	Log	Quadratic
	rep(1,17)	6.803	0.735	0.996	0.673
18	c(1:18)	156.583	0.706	0.994	0.645
	rep(1,18)	6.967	0.756	0.997	0.690
19	c(1:19)	172.858	0.720	0.995	0.650
	rep(1,19)	7.167	0.769	0.998	0.698
20	c(1:20)	189.306	0.738	0.997	0.669
	rep(1,20)	7.373	0.781	0.999	0.712
21	c(1:21)	207.243	0.748	0.997	0.677
	rep(1,21)	7.586	0.792	0.999	0.720
22	c(1:22)	224.638	0.766	0.998	0.688
	rep(1,22)	7.669	0.813	0.999	0.734
23	c(1:23)	245.186	0.768	0.998	0.684
	rep(1,23)	7.855	0.822	0.999	0.747
24	c(1:24)	263.494	0.784	0.999	0.699
	rep(1,24)	8.054	0.830	1.000	0.750
25	c(1:25)	284.035	0.796	0.999	0.708
	rep(1,25)	8.173	0.845	1.000	0.759
26	c(1:26)	305.435	0.800	0.999	0.710
	rep(1,26)	8.386	0.848	1.000	0.766
27	c(1:27)	326.062	0.813	1.000	0.722
	rep(1,27)	8.553	0.855	1.000	0.768
28	c(1:28)	349.506	0.819	1.000	0.724
	rep(1,28)	8.666	0.868	1.000	0.776
29	c(1:29)	373.453	0.823	1.000	0.724
	rep(1,29)	8.842	0.872	1.000	0.781
30	c(1:30)	394.562	0.838	1.000	0.738
	rep(1,30)	8.946	0.880	1.000	0.789
31	c(1:31)	418.801	0.843	1.000	0.742
	rep(1,31)	9.136	0.885	1.000	0.789
32	c(1:32)	443.617	0.851	1.000	0.748
	rep(1,32)	9.331	0.888	1.000	0.787
33	c(1:33)	469.796	0.857	1.000	0.745
	rep(1,33)	9.408	0.895	1.000	0.795
34	c(1:34)	497.823	0.860	1.000	0.745
	rep(1,34)	9.581	0.898	1.000	0.796

Continued

Algorithm 2					
			Power		
n	Weights	Critical value	Linear	Log	Quadratic
35	c(1:35)	524.230	0.864	1.000	0.750
	rep(1,35)	9.691	0.903	1.000	0.801
36	c(1:36)	549.232	0.875	1.000	0.760
	rep(1,36)	9.900	0.908	1.000	0.798
37	c(1:37)	579.489	0.875	1.000	0.757
	rep(1,37)	10.048	0.909	1.000	0.803
38	c(1:38)	608.407	0.884	1.000	0.759
	rep(1,38)	10.057	0.917	1.000	0.808
39	c(1:39)	638.066	0.885	1.000	0.761
	rep(1,39)	10.233	0.917	1.000	0.805
40	c(1:40)	668.788	0.886	1.000	0.762
	rep(1,40)	10.407	0.920	1.000	0.806
41	c(1:41)	701.636	0.887	1.000	0.760
	rep(1,41)	10.516	0.926	1.000	0.806
42	c(1:42)	731.158	0.894	1.000	0.763
	rep(1,42)	10.587	0.927	1.000	0.809
43	c(1:43)	764.307	0.896	1.000	0.766
	rep(1,43)	10.828	0.929	1.000	0.808
44	c(1:44)	799.082	0.896	1.000	0.761
	rep(1,44)	10.875	0.933	1.000	0.807
45	c(1:45)	831.269	0.900	1.000	0.766
	rep(1,45)	11.018	0.933	1.000	0.811
46	c(1:46)	865.438	0.904	1.000	0.766
	rep(1,46)	11.140	0.935	1.000	0.806
47	c(1:47)	902.158	0.904	1.000	0.758
	rep(1,47)	11.275	0.935	1.000	0.809
48	c(1:48)	935.650	0.911	1.000	0.766
	rep(1,48)	11.377	0.937	1.000	0.808
49	c(1:49)	974.517	0.908	1.000	0.761
	rep(1,49)	11.496	0.938	1.000	0.804
50	c(1:50)	1010.099	0.911	1.000	0.763
	rep(1,50)	11.640	0.940	1.000	0.807

Chapter 5

Illustrative Example

In this chapter, we present a real-life example to illustrate the proposed test procedure.

5.1 Ozone Data

For the purpose of comparison, we use the data set presented in Hofmann and Balakrishnan (2006), which is originally from Box *et al.* (1994). Figure 5.1 presents the data set of monthly averages of hourly ozone readings in downtown Los Angeles during the period 1955-1972. It exhibits an overall decreasing trend obscured by seasonal effects. The complete data set is given in Appendix B.

When using the complete data set, all test statistics in Hofmann and Balakrishnan (2006) detected the trend with p-values < 0.001. The same thing happens to the test statistic proposed here. So it is hard to compare these procedures in this case. In order to make a clear comparison, it is necessary to assess the performance of the tests on the shorter intervals of consecutive observations where the trend is harder to identify. Then, the consecutive k-year intervals ($k = 4, \ . \ . \ . \ , 10$) are used from the 18 years of data, each starting in January. Thus there are $19 - k$ such intervals starting in years $1955, 1956, \ . \ . \ . \ , 1973 - k$.

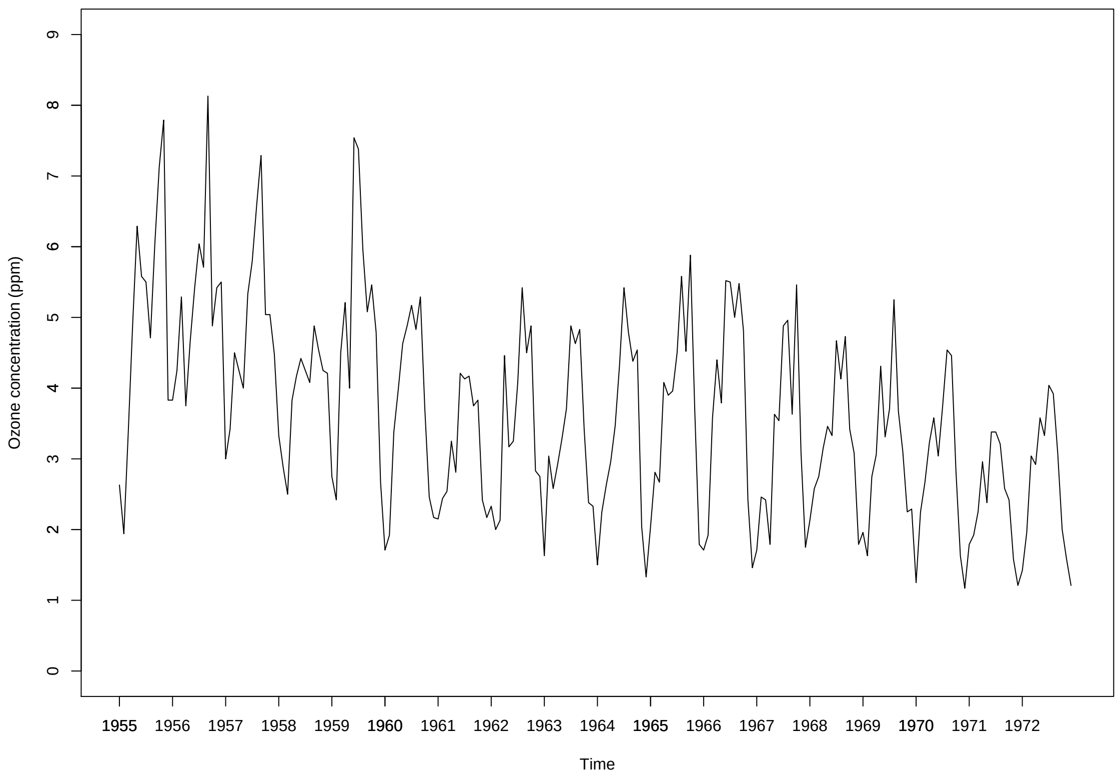

Figure 5.1: Monthly averages of hourly ozone readings in downtown Los Angeles from 1955 to 1972.

5.2 Data Transformation

Before applying the test procedure, we need to make an adjustment to the raw data in accordance with the null distribution by the following standardization.

For every k ($k = 4, 5, ..., 10$) and each interval, we transform the data by

$$y_{ji}{}^{*} = \frac{y_{ji} - mean_{1i}}{sd_i}, \qquad \text{for } n = 12, \tag{5.1}$$

and

$$y_{ji}{}^{**} = \frac{y_{ji} - (mean_{1i} + mean_{2i})/2}{sd_i}, \qquad \text{for } n = 24, \tag{5.2}$$

respectively, where $i = 1, 2, ..., 19 - k$, $j = 1, 2, ..., 12k$, y_{ji} is the j-th observation of the i-th interval, $mean_{1i}$ and $mean_{2i}$ are the means of the first year and the second year of the i-th interval, respectively, sd_i is the standard deviation of the i-th interval. Thus, for each interval, the observations are adjusted to have a standard deviation of 1; if we choose $n = 12$, the observations of the first year are adjusted to have a mean of 0 by (5.1); if we choose $n = 24$, the observations of the first and second years are adjusted to have a mean of 0 by (5.2).

5.3 Median p-values

We use the weights $\boldsymbol{c}' = (1, 1, \ldots, 1)$ and the adjusted observations of last one and two years in each interval to construct our test statistics. For each k, we get the median test statistic value over all intervals after calculating the test statistic values in each interval, which is the summation of the last 12 or 24 standardized values depending on the sample size we choose. Then, we obtain the median p-value over all intervals by finding $P(I <$ the median test statistic value) in $25,000$ simulations using Algorithm 2. The median p-value is a robust measure which can clearly describe the behavior of the test statistic in shorter intervals.

For each k ($k = 4, 5, ..., 10$), all the median test statistic values and the median p-values are summarized in Table 5.1. These results do not reject a strong decreasing trend in the data. The median p-values are less than 0.05 except for the cases of k=4 and 5 when $n = 12$ and the case of k=4 when $n = 24$.

Compared with the tests in Hofmann and Balakrishnan (2006) (see Appendix B Table B.3), the proposed test statistic shows smaller p values in 4 out of 7 cases for $n = 12$ and smaller p values in all the cases for $n = 24$, and hence appears to be better able to detect the trend in the latter case for this data set. Note that we only use at most one-quarter of the data for $n = 12$ and one-half of the data for $n = 24$ in each interval to calculate the values of the test statistic.

Table 5.1: Median p-values for the $19 - k$ possibilities of consecutive k-year intervals

Subset length in years	Number of ozone readings	Number of k-year intervals	Median test statistic values		Median p-values	
(k)	($n = 12k$)	($19 - k$)	$n = 12$	$n = 24$	$n = 12$	$n = 24$
4	48	15	-4.8539	-6.8757	0.0823	0.0807
5	60	14	-5.4488	-12.1526	0.0601	0.0065
6	72	13	-6.9073	-15.6992	0.0227	0.0006
7	84	12	-7.6832	-15.7419	0.0130	0.0005
8	96	11	-8.6225	-15.8332	0.0060	0.0005
9	108	10	-7.5134	-12.8800	0.0150	0.0044
10	120	9	-9.8368	-15.2024	0.0023	0.0010

Chapter 6

Concluding Remarks

6.1 Summary

The investigation of trends is an important issue in many applications. We are interested in testing for the presence of monotone trends in a time series model . The observation in the model is considered to be the sum of two independent components: the trend and the Gaussian white noise. The trend component is the multiplication of a trend coefficient and a trend function. A positive (negative) trend coefficient represents an increasing (decreasing) trend.

We propose an exact test procedure. The test statistic we propose is the linear combination of order statistics of last moving sample. Our derivation shows that such test statistic follows the closed skew normal (CSN) distribution under the null hypothesis. The partitioned-conditional method shows that a CSN random vector is distributionally equivalent to a normal random vector with several components that are greater than a given vector. Furthermore, the CSN vector can be decomposed into a combination of two independent components: a truncated normal vector and a standard normal vector. Based on the decomposition, two algorithms are presented and compared. By generating the truncated normal vector directly from a multivariate truncated distribution, the second algorithm is found to be much more efficient in terms of time than the first algorithm when the sample size increases. After examining the empirical Type I error rates, we evaluate the constructed test procedure with linear, logarithmic and quadratic trend, respectively, by an empirical power study. The test can detect the trends with large power values for a small data set. When dealing with a big data set, the test performs consistently best with log trend and worst under quadratic trend. Compared with the nonparametric tests in Hofmann and Balakrishnan (2006), which require the whole data set, the proposed test can achieve much higher power by using only one-quarter of the full data set when the normal distribution assumption is satisfied. Note that, if we don't know what distribution we are working with, or we are dealing with non normal data, it will be more beneficial to use the distribution-free tests in Hofmann and Balakrishnan (2006), which can work for any type of distribution.

6.2 Future Work

We may consider the robustness of the test in two ways. Firstly, we could explore the performance of the test under different distributions, such as skew normal and elliptical distribution. Secondly, we could examine the performance of the test in presence of outliers or correlation.

The exact distribution under the alternative hypothesis will be of interest to study. Another problem of interest will be to determine an optimal test statistic in the situation considered here rather than fixing the coefficients of the L-statistic in an ad-hoc way. We hope to consider these issues for further study.

Appendix A

R code

A.1 R code for Algorithm 1

A.1.1 R code for runtime of Algorithm 1 in Table 4.1

```
CSN <- function(Mu,Sigma,Gamma,Nu,Delta,num){

  n2=length(Mu)

  n1=length(Nu)

  Gamma_vt=Gamma%*%Sigma

  Sigma_v=Delta+Gamma%*%Sigma%*%t(Gamma)

  Fm=t(Gamma_vt)%*%solve(Sigma_v)

  Gm=Sigma-Fm%*%Gamma_vt

  Sigma_v_chol=t(chol(Sigma_v))

  Gm_chol=t(chol(Gm))

  res=matrix(0,num,n2)

  i=0

  k=0
```

```
  while (i < num) {
    i=i+1
    z1=matrix(rnorm(n1),ncol=1)
    v=Sigma_v_chol%*%z1-Nu
    while (any(v<0))  {
      k=k+1
      z1=matrix(rnorm(n1),ncol=1)
      v=Sigma_v_chol%*%z1-Nu
    }
    z2=matrix(rnorm(n2),ncol=1)
    res[i,]=Mu+Fm%*%(v+Nu)+Gm_chol%*%z2
  }
  return(res)
}
for (n in 2:9){
  W=1
  Mu=(c(0))
  a=c(1:n)
  A=W*a
  Sigma=t(A)%*%A
  B=matrix(0,n-1,n)
  B[1,]=c(-1,1,rep(0,n-2))
  for (i in 2:n-1) {
    B[i,]=c(rep(0,i-1),-1,1,rep(0,n-i-1))
  }
  Gamma=B%*%A%*%solve(t(A)%*%A)
```

```
  Nu=rep(0,n-1)
  Delta=B%*%t(B)-B%*%A%*%solve(t(A)%*%A)%*%t(A)%*%t(B)
  print(system.time(myCSN<-CSN(Mu,Sigma,Gamma,Nu,Delta,1000)))
}
```

A.1.2 R code for critical values and power values by Algorithm 1 in Table 4.4

```
CSN <- function(Mu,Sigma,Gamma,Nu,Delta,num){
  n2=length(Mu)
  n1=length(Nu)
  Gamma_vt=Gamma%*%Sigma
  Sigma_v=Delta+Gamma%*%Sigma%*%t(Gamma)
  Fm=t(Gamma_vt)%*%solve(Sigma_v)
  Gm=Sigma-Fm%*%Gamma_vt
  Sigma_v_chol=t(chol(Sigma_v))
  Gm_chol=t(chol(Gm))
  res=matrix(0,num,n2)
  i=0
  k=0
  while (i < num) {
    i=i+1
    z1=matrix(rnorm(n1),ncol=1)
    v=Sigma_v_chol%*%z1-Nu
    while (any(v<0))  {
      k=k+1
```

```
      z1=matrix(rnorm(n1),ncol=1)
      v=Sigma_v_chol%*%z1-Nu
    }
    z2=matrix(rnorm(n2),ncol=1)
    res[i,]=Mu+Fm%*%(v+Nu)+Gm_chol%*%z2
  }
  w=0.95*num
  return(sort(res)[w])
}
cr=rep(0,n-1)
for (n in 2:5){
  W=1
  Mu=(c(0))
  a=c(1:n)##change the weight to "a=rep(1,n)"
  A=W*a
  Sigma=t(A)%*%A
  B=matrix(0,n-1,n)
  B[1,]=c(-1,1,rep(0,n-2))
  for (i in 2:n-1) {
    B[i,]=c(rep(0,i-1),-1,1,rep(0,n-i-1))
  }
  Gamma=B%*%A%*%solve(t(A)%*%A)
  Nu=rep(0,n-1)
  Delta=B%*%t(B)-B%*%A%*%solve(t(A)%*%A)%*%t(A)%*%t(B)
  cr[n-1]=CSN(Mu,Sigma,Gamma,Nu,Delta,1000000)
}
```

```
print(cr)

alpha=function(n,W,beta,cr,m,asim) {
  a=c(1:n)##change the weight to "a=rep(1,n)"
  A=W*a
  y=array(0,c(m,asim))
  M=array(0,c(n,asim))
  MS=array(0,c(n,asim))
  MM=array(0,c(asim))
  Time=c(1:m)
  Trend=beta*Time##change the form of trend to "Trend=beta*log(Time)"
               ## and "Trend=beta*(Time^2+beta*Time)", respectively.
  for(j in 1:asim){
    y[,j]=rnorm(m)+Trend
  }
  M=y[(m-n+1):m,]
  MS=apply(M, 2, sort)
  MM=A%*%MS
  power=length(MM[MM >= cr ])/asim
  return(power)
}
beta=0.25##change the value of beta to 1.2 and 0.023, respectively
power=rep(0,n-1)
for (n in 2:5){
  power[n-1]=alpha(n,1,beta,cr[n-1],10,1000000)
}
print(power)
```

A.2 R code for Algorithm 2

A.2.1 R code for runtime of Algorithm 2 in Table 4.1

```
library("tmvtnorm")

CSN <- function(Mu,Sigma,Gamma,Nu,Delta,num){

  n2=length(Mu)

  n1=length(Nu)

  Gamma_vt=Gamma%*%Sigma

  Sigma_v=Delta+Gamma%*%Sigma%*%t(Gamma)

  Fm=t(Gamma_vt)%*%solve(Sigma_v)

  Gm=Sigma-Fm%*%Gamma_vt

  Gm_chol=t(chol(Gm))

  res=matrix(0,num,n2)

  i=0

  while (i < num) {

    i=i+1

    v <- rtmvnorm(n=1, mean=rep(0,n1),sigma=Sigma_v, lower=rep(0,n1),

                  algorithm="gibbs", burn.in.samples=100)

    ##need to change "sigma=Sigma_v" to "sigma=(Sigma_v)^{1/2}" when n=2

    u=matrix(rnorm(n2),ncol=1)

    res[i,]=Mu+Fm%*%(t(v)+Nu)+Gm_chol%*%u

  }

  return(res)

}

for (n in 2:9){

  W=1
```

```
  Mu=(c(0))
  a=c(1:n)
  A=W*a
  Sigma=t(A)%*%A
  B=matrix(0,n-1,n)
  B[1,]=c(-1,1,rep(0,n-2))
  for (i in 2:n-1) {
    B[i,]=c(rep(0,i-1),-1,1,rep(0,n-i-1))
  }
  Gamma=B%*%A%*%solve(t(A)%*%A)
  Nu=rep(0,n-1)
  Delta=B%*%t(B)-B%*%A%*%solve(t(A)%*%A)%*%t(A)%*%t(B)
  print(system.time(myCSN<-CSN(Mu,Sigma,Gamma,Nu,Delta,1000)))
}
```

A.2.2 R code for critical values and power values by Algorithm 2 in Table 4.4

```
library("tmvtnorm")
CSN <- function(Mu,Sigma,Gamma,Nu,Delta,num){
  n2=length(Mu)
  n1=length(Nu)
  Gamma_vt=Gamma%*%Sigma
  Sigma_v=Delta+Gamma%*%Sigma%*%t(Gamma)
  Fm=t(Gamma_vt)%*%solve(Sigma_v)
  Gm=Sigma-Fm%*%Gamma_vt
```

```
  Gm_chol=t(chol(Gm))
  res=matrix(0,num,n2)
  i=0
  while (i < num) {
    i=i+1
    v <- rtmvnorm(n=1, mean=rep(0,n1),sigma=Sigma_v, lower=rep(0,n1),
                  algorithm="gibbs", burn.in.samples=100)
    ##need to change "sigma=Sigma_v" to "sigma=(Sigma_v)^{1/2}" when n=2
    u=matrix(rnorm(n2),ncol=1)
    res[i,]=Mu+Fm%*%(t(v)+Nu)+Gm_chol%*%u
  }
  w=0.95*num
  return(sort(res)[w])
}
cr=rep(0,n-1)
for (n in 2:5){
  W=1
  Mu=(c(0))
  a=c(1:n)##change the weight to "a=rep(1,n)"
  A=W*a
  Sigma=t(A)%*%A
  B=matrix(0,n-1,n)
  B[1,]=c(-1,1,rep(0,n-2))
  for (i in 2:n-1) {
    B[i,]=c(rep(0,i-1),-1,1,rep(0,n-i-1))
  }
```

```
  Gamma=B%*%A%*%solve(t(A)%*%A)
  Nu=rep(0,n-1)
  Delta=B%*%t(B)-B%*%A%*%solve(t(A)%*%A)%*%t(A)%*%t(B)
  cr[n-1]=CSN(Mu,Sigma,Gamma,Nu,Delta,1000000)
}
print(cr)
alpha=function(n,W,beta,cr,m,asim) {
  a=c(1:n)##change the weight to "a=rep(1,n)"
  A=W*a
  y=array(0,c(m,asim))
  M=array(0,c(n,asim))
  MS=array(0,c(n,asim))
  MM=array(0,c(asim))
  Time=c(1:m)
  Trend=beta*Time##change the form of trend to "Trend=beta*log(Time)"
               ## and "Trend=beta*(Time^2+beta*Time)", respectively.
  for(j in 1:asim){
    y[,j]=rnorm(m)+Trend
  }
  M=y[(m-n+1):m,]
  MS=apply(M, 2, sort)
  MM=A%*%MS
  power=length(MM[MM >= cr ])/asim
  return(power)
}
beta=0.25##change the value of beta to 1.2 and 0.023, respectively
```

```
power=rep(0,n-1)
for (n in 2:5){
  power[n-1]=alpha(n,1,beta,cr[n-1],10,1000000)
}
print(power)
```

A.2.3 R code for critical values and power values by Algorithm 2 in Table 4.5

```
CSN.sim<-function(Mu,Sigma,Gamma,Nu,Delta,n.samples=58000,n.bis=100,
                  max.iter=n.samples*10){
  require(tmvtnorm,quietly=TRUE)
  n2 <- length(Mu)
  n1 <- length(Nu)
  Gamma_vt <- Gamma%*%Sigma
  Sigma_v <- Delta+Gamma%*%Sigma%*%t(Gamma)
  Fm <- t(Gamma_vt)%*%solve(Sigma_v)
  Gm <- Sigma-Fm%*%Gamma_vt
  Gm_chol <- t(chol(Gm))
  res <- matrix(0,n.samples,n2)
  i<-0;j<-0
  while (i < n.samples&j<max.iter){
    v <- rtmvnorm(n=1, mean=rep(0,n1),sigma=Sigma_v, lower=rep(0,n1),
                  algorithm="gibbs", burn.in.samples=n.bis)
     ##need to change "sigma=Sigma_v" to "sigma=(Sigma_v)^{1/2}" when n=2
    if(!any(is.na(v))){
```

```
      i<-i+1
      u=matrix(rnorm(n2),ncol=1)
      res[i,]<- Mu+Fm%*%(t(v)+Nu)+Gm_chol%*%u
    }
    j<-j+1
  }
  if(j==max.iter&i<n.samples){print("Required number of samples not achieved")}
  print(c(i,j))
  w=0.95*n.samples
  return(sort(res)[w])
}
cr=rep(0,n-1)
for (n in 2:50){
  W=1
  Mu=(c(0))
  a=c(1:n)##change the weight to "a=rep(1,n)"
  A=W*a
  Sigma=t(A)%*%A
  B=matrix(0,n-1,n)
  B[1,]=c(-1,1,rep(0,n-2))
  for (i in 2:n-1) {
    B[i,]=c(rep(0,i-1),-1,1,rep(0,n-i-1))
  }
  Gamma=B%*%A%*%solve(t(A)%*%A)
  Nu=rep(0,n-1)
  Delta=B%*%t(B)-B%*%A%*%solve(t(A)%*%A)%*%t(A)%*%t(B)
```

```
  cr[n-1]=CSN.sim(Mu,Sigma,Gamma,Nu,Delta,n.samples=58000,n.bis=100)
}
print(cr)
alpha=function(n,W,beta,cr,m,asim) {
  a=c(1:n)##change the weight to "a=rep(1,n)"
  A=W*a
  y=array(0,c(m,asim))
  M=array(0,c(n,asim))
  MS=array(0,c(n,asim))
  MM=array(0,c(asim))
  Time=c(1:m)
  Trend=beta*Time##change the form of trend to "Trend=beta*log(Time)"
  ## and "Trend=beta*(Time^2+beta*Time)", respectively.
  for(j in 1:asim){
    y[,j]=rnorm(m)+Trend
  }
  M=y[(m-n+1):m,]
  MS=apply(M, 2, sort)
  MM=A%*%MS
  power=length(MM[MM >= cr ])/asim
  return(power)
}
beta=0.006##change the value of beta to 0.23 and 0.00006, respectively
power=rep(0,n-1)
for (n in 2:50){
  power[n-1]=alpha(n,1,beta,cr[n-1],100,58000)
```

```
}
print(power)
```

A.2.4 R code for median p-values by Algorithm 2 in Table 5.1

```
mtest=c(-4.853903,-5.448766,-6.907302,-7.683202,-8.622535,-7.513401,-9.836821)
library("tmvtnorm")
CSN <- function(Mu,Sigma,Gamma,Nu,Delta,num){
  n2=length(Mu)
  n1=length(Nu)
  Gamma_vt=Gamma%*%Sigma
  Sigma_v=Delta+Gamma%*%Sigma%*%t(Gamma)
  Fm=t(Gamma_vt)%*%solve(Sigma_v)
  Gm=Sigma-Fm%*%Gamma_vt
  Gm_chol=t(chol(Gm))
  res=matrix(0,num,n2)
  i=0
  while (i < num) {
    i=i+1
    v <- rtmvnorm(n=1, mean=rep(0,n1),sigma=Sigma_v, lower=rep(0,n1),
                  algorithm="gibbs", burn.in.samples=100)
    u=matrix(rnorm(n2),ncol=1)
    res[i,]=Mu+Fm%*%(t(v)+Nu)+Gm_chol%*%u
  }
  return(list(mean(res< mtest[1]),mean(res< mtest[2]),mean(res< mtest[3])
```

```
              , mean(res< mtest[4]),  mean(res< mtest[5])
              , mean(res< mtest[6]), mean(res< mtest[7])))
}
n=12 ##change n to 24 for n=24
W=1
Mu=(c(0))
a=rep(1,n)
A=W*a
Sigma=t(A)%*%A
B=matrix(0,n-1,n)
B[1,]=c(-1,1,rep(0,n-2))
for (i in 2:n-1) {
  B[i,]=c(rep(0,i-1),-1,1,rep(0,n-i-1))
}
Gamma=B%*%A%*%solve(t(A)%*%A)
Nu=rep(0,n-1)
Delta=B%*%t(B)-B%*%A%*%solve(t(A)%*%A)%*%t(A)%*%t(B)
p=unlist(CSN(Mu,Sigma,Gamma,Nu,Delta,25000))
print(p)
```

A.3 R code for median test statistic values in Table 5.1

```
ozone=c(2.63, 1.94, 3.38, 4.92, 6.29, 5.58, 5.50, 4.71, 6.04, 7.13, 7.79, 3.83,
        3.83, 4.25, 5.29, 3.75, 4.67, 5.42, 6.04, 5.71, 8.13, 4.88, 5.42, 5.50,
```

```
        3.00, 3.42, 4.50, 4.25, 4.00, 5.33, 5.79, 6.58, 7.29, 5.04, 5.04, 4.48,
        3.33, 2.88, 2.50, 3.83, 4.17, 4.42, 4.25, 4.08, 4.88, 4.54, 4.25, 4.21,
        2.75, 2.42, 4.50, 5.21, 4.00, 7.54, 7.38, 5.96, 5.08, 5.46, 4.79, 2.67,
        1.71, 1.92, 3.38, 3.98, 4.63, 4.88, 5.17, 4.83, 5.29, 3.71, 2.46, 2.17,
        2.15, 2.44, 2.54, 3.25, 2.81, 4.21, 4.13, 4.17, 3.75, 3.83, 2.42, 2.17,
        2.33, 2.00, 2.13, 4.46, 3.17, 3.25, 4.08, 5.42, 4.50, 4.88, 2.83, 2.75,
        1.63, 3.04, 2.58, 2.92, 3.29, 3.71, 4.88, 4.63, 4.83, 3.42, 2.38, 2.33,
        1.50, 2.25, 2.63, 2.96, 3.46, 4.33, 5.42, 4.79, 4.38, 4.54, 2.04, 1.33,
        2.04, 2.81, 2.67, 4.08, 3.90, 3.96, 4.50, 5.58, 4.52, 5.88, 3.67, 1.79,
        1.71, 1.92, 3.58, 4.40, 3.79, 5.52, 5.50, 5.00, 5.48, 4.81, 2.42, 1.46,
        1.71, 2.46, 2.42, 1.79, 3.63, 3.54, 4.88, 4.96, 3.63, 5.46, 3.08, 1.75,
        2.13, 2.58, 2.75, 3.15, 3.46, 3.33, 4.67, 4.13, 4.73, 3.42, 3.08, 1.79,
        1.96, 1.63, 2.75, 3.06, 4.31, 3.31, 3.71, 5.25, 3.67, 3.10, 2.25, 2.29,
        1.25, 2.25, 2.67, 3.23, 3.58, 3.04, 3.75, 4.54, 4.46, 2.83, 1.63, 1.17,
        1.79, 1.92, 2.25, 2.96, 2.38, 3.38, 3.38, 3.21, 2.58, 2.42, 1.58, 1.21,
        1.42, 1.96, 3.04, 2.92, 3.58, 3.33, 4.04, 3.92, 3.08, 2.00, 1.58, 1.21)
mean1=apply(matrix(ozone,12,18),2,mean)
mtest=array(0,7)
for (k in (4:10)){
  ozone.k=array(0,c((12*k),(19-k),k))
  ozone.k.s=array(0,c((12*k),(19-k),k))
  for (i in 1:(19-k)){
    ozone.k[,i,k]=ozone[(12*(i-1)+1):(12*(k+(i-1)))]
    ozone.k.s[,i,k]=(ozone.k[,i,k]-mean1[i])/sd(ozone.k[,i,k])
    ##change the above equation to
    ##ozone.k.s[,i,k]=(ozone.k[,i,k]-mean1[i])/sd(ozone.k[,i,k]) for n=24
```

```
  }
  fsum <- function(x) c(sum = sum(x[(12*(k-1)+1):(12*k)]))
  ##change the above function to
  ##fsum <- function(x) c(sum = sum(x[(12*(k-2)+1):(12*k)])) for n=24
test=array(0,19-k)
test=apply(ozone.k.s[,,k],2,fsum)
mtest[k-3]=median(test)
}
print(mtest)
```

Appendix B

Tables

B.1 Tables

B.1.1 Power of the nonparametric tests in Hofmann and Balakrishnan (2006)

B.1.2 Monthly averages of hourly readings of ozone in downtown Los Angeles

B.1.3 Median p-values for the $(19-k)$ possibilities of consecutive k-year intervals in Hofmann and Balakrishnan (2006)

Table B.1: Power of the nonparametric tests in Hofmann and Balakrishnan (2006)

			Power					
			Q	r_S	D	B	M	$G(0.995)$
m	Trend	β	0.036	0.048	0.043	0.047	0.050	0.050
10	Linear	0.25	0.545	0.619	0.418	0.467	0.632	0.624
	Log	1.2	0.586	0.649	0.502	0.47	0.666	0.674
	Quadratic	0.023	0.539	0.612	0.415	0.455	0.626	0.62
			Power					
			Q	r_S	D	B	M	$G(0.995)$
m	Trend	β	0.050	0.050	0.044	0.050	0.050	0.050
100	Linear	0.006	0.52	0.52	0.13	0.39	0.53	0.52
	Log	0.23	0.58	0.58	0.2	0.44	0.59	0.6
	Quadratic	0.00006	0.52	0.53	0.13	0.4	0.54	0.53

Note: Q,r,D,B,M and G are the nonparametric test statistics presented in Hofmann and Balakrishnan (2006).

Table B.2: Monthly averages of hourly readings of ozone in downtown Los Angeles.

	Jan.	Feb.	Mar.	Apr.	May	June	July	Aug.	Sept.	Oct.	Nov.	Dec.
1955	2.63	1.94	3.38	4.92	6.29	5.58	5.50	4.71	6.04	7.13	7.79	3.83
1956	3.83	4.25	5.29	3.75	4.67	5.42	6.04	5.71	8.13	4.88	5.42	5.50
1957	3.00	3.42	4.50	4.25	4.00	5.33	5.79	6.58	7.29	5.04	5.04	4.48
1958	3.33	2.88	2.50	3.83	4.17	4.42	4.25	4.08	4.88	4.54	4.25	4.21
1959	2.75	2.42	4.50	5.21	4.00	7.54	7.38	5.96	5.08	5.46	4.79	2.67
1960	1.71	1.92	3.38	3.98	4.63	4.88	5.17	4.83	5.29	3.71	2.46	2.17
1961	2.15	2.44	2.54	3.25	2.81	4.21	4.13	4.17	3.75	3.83	2.42	2.17
1962	2.33	2.00	2.13	4.46	3.17	3.25	4.08	5.42	4.50	4.88	2.83	2.75
1963	1.63	3.04	2.58	2.92	3.29	3.71	4.88	4.63	4.83	3.42	2.38	2.33
1964	1.50	2.25	2.63	2.96	3.46	4.33	5.42	4.79	4.38	4.54	2.04	1.33
1965	2.04	2.81	2.67	4.08	3.90	3.96	4.50	5.58	4.52	5.88	3.67	1.79
1966	1.71	1.92	3.58	4.40	3.79	5.52	5.50	5.00	5.48	4.81	2.42	1.46
1967	1.71	2.46	2.42	1.79	3.63	3.54	4.88	4.96	3.63	5.46	3.08	1.75
1968	2.13	2.58	2.75	3.15	3.46	3.33	4.67	4.13	4.73	3.42	3.08	1.79
1969	1.96	1.63	2.75	3.06	4.31	3.31	3.71	5.25	3.67	3.10	2.25	2.29
1970	1.25	2.25	2.67	3.23	3.58	3.04	3.75	4.54	4.46	2.83	1.63	1.17
1971	1.79	1.92	2.25	2.96	2.38	3.38	3.38	3.21	2.58	2.42	1.58	1.21
1972	1.42	1.96	3.04	2.92	3.58	3.33	4.04	3.92	3.08	2.00	1.58	1.21

Note: 216 observation; values are in pphm (Box *et al.*, 1994).

Table B.3: Median p-values for the $(19-k)$ possibilities of consecutive k-year intervals in Hofmann and Balakrishnan (2006)

Subset length in years (k)	Number of ozone readings ($n = 12k$)	Number of k-year intervals ($19-k$)	Q	r_S	D	B	M	G (0.99)	G (0.995)	G (0.999)
4	48	15	0.277	0.256	0.269	0.215	0.361	0.182	0.173	0.172
5	60	14	0.112	0.095	0.198	0.072	0.167	0.082	0.079	0.075
6	72	13	0.037	0.033	0.279	0.019	0.09	0.023	0.022	0.018
7	84	12	0.069	0.062	0.102	0.042	0.051	0.021	0.016	0.008
8	96	11	0.086	0.071	0.042	0.05	0.064	0.044	0.036	0.023
9	108	10	0.094	0.091	0.037	0.055	0.089	0.028	0.023	0.014
10	120	9	0.017	0.017	0.03	0.011	0.049	0.008	0.007	0.006

Note: Q,r,D,B,M and G are the nonparametric test statistics presented in Hofmann and Balakrishnan (2006).

Bibliography

Allard, D. and Naveau, P. (2007). A new spatial skew-normal random field model. *Communications in Statistics:Theory and Methods*, **36**(9), 1821–1834.

Azzalini, A. (1985). A class of distributions which includes the normal ones. *Scandinavian Journal of Statistics*, **12**(2), 171–178.

Azzalini, A. and Dalla Valle, A. (1996). The multivariate skew-normal distribution. *Biometrika*, **83**(4), 715–726.

Balakrishnan, N. and Cohen, A. C. (1991). *Order Statistics and Inference: Estimation Methods.* Academic Press, Boston.

Box, G. E. P., Jenkins, G. M., and Reinsel, G. C. (1994). *Time Series Analysis: Forecasting and Control,* 3rd edition. Prentice Hall, New Jersey.

Chatfield, C. (2003). *The Analysis of Time Series: An Introduction,* 6th edition. Chapman & Hall/CRC Press, Boca Raton, FL.

Cleveland, W. S. and Kleiner, B. (1975). A graphical technique for enhancing scatterplots with moving statistics. *Technometrics*, **17**(4), 447–454.

Crocetta, C. and Loperfido, N. (2005). The exact sampling distribution of L-statistics. *Metron*, **LXIII**, 213–223.

David, H. A. and Nagaraja, H. N. (2003). *Order Statistics,* 3rd edition. John Wiley & Sons, Hoboken, New Jersey.

David, H. A. and Rogers, M. P. (1983). Order statistics in overlapping samples, moving order statistics and U-statistics. *Biometrika*, **70**(1), 245–249.

Domínguez-Molina, J. A., González-Farías, G., and Gupta, A. K. (2003). The multivariate closed skew normal distribution. Technical Report No. 03-12, Department of Mathematics and Statistics, Bowling Green State University, Bowling Green, Ohio.

González-Farías, G., Domínguez-Molina, J. A., and Gupta, A. K. (2004a). Additive properties of skew normal random vectors. *Journal of Statistical Planning and Inference*, **126**(2), 521–534.

González-Farías, G., Domínguez-Molina, J. A., and Gupta, A. K. (2004b). The closed skew-normal distribution. In *Skew-Elliptical Distributions and Their Applications: A Journey Beyond Normality* (Ed., M. G. Genton), Chapter 2, pp. 25–42. Chapman & Hall/CRC Press, Boca Raton, FL.

Gupta, A. K., González-Farías, G., and Domínguez-Molina, J. A. (2004). A multivariate skew normal distribution. *Journal of Multivariate Analysis*, **89**(1), 181–190.

Hofmann, G. and Balakrishnan, N. (2006). A nonparametric test for trend based on initial ranks. *Journal of Statistical Computation and Simulation*, **76**(9), 829–837.

Inagaki, N. (1980). The distribution of moving order statistics. In *Recent Developments in Statistical Inference and Data Analysis,* pp. 137–142.

Iversen, D. H. (2010). *Closed-skew Distributions: Simulation, Inversion and Parameter Estimation*. Master's thesis, Norwegian University of Science and Technology.

Johnson, R. A. and Wichern, D. W. (2007). *Applied Multivariate Statistical Analysis*. Prentice Hall, New York.

Meko, D. (2013). *Applied Time Series Analysis*. http://www.ltrr.arizona.edu/~dmeko/geos585a.html/. Last visited: 09/02/2013.