Development of Penalized Complexity Priors for Stationary and Invertible Time Series Processes

Himanshu Srivastav

Master of Science in Mathematical Sciences
Submission date: June 2017
Supervisor: Jo Eidsvik, IMF

Norwegian University of Science and Technology
Department of Mathematical Sciences

Development of Penalized Complexity Priors for Stationary and Invertible Time Series Processes

Himanshu Srivastav

June 6, 2017

Contents

Preface

This report is the result of my master thesis in MA3911 at NTNU, which was conducted during the period between September 2016 and June 2017. The statistics program involves learning of the time series models and computer intensive methods for statistical inference. The curriculum of TMA4300 - Computer Intensive Statistical Methods gave me an introduction about the INLA framework. When I approached Prof Håvard Rue to know more about the INLA framework and research work in this field, I got an opportunity to work with him on a studforsk project. The topic of studforsk project was "Small Study on the Penalized Complexity Priors for stationary auto regressive (AR) for order 1". After completing the studforsk project, I decided to extend this work for other time series models as my master thesis.

Acknowledgement

Looking back on my time in NTNU, I would like to thank to my professors. I was constantly impressed by their strict practical scientific attitude and progressive spirit. I am thankful to my supervisor Prof Jo Eidsvik for giving me a great guidance and support to finalize my thesis. He provided me with a lot of useful information needed for the thesis and gave a lot of valuable advice on the problems encountered in the process. I express my deep respect to him. I am also grateful to Prof Håvard Rue for helping me to select this topic and also for providing critical inputs every now and then. His constant guidance made it possible for me to complete the thesis. At the end, I would like to thank my parents and my friends for their support. Their love, understanding and patience give me great encouragement.

Abstract

The autoregressive process of order 1 (AR(1)), moving average process of order 1 (MA(1)) and autoregressive moving average process of order $(1,1)$ (ARMA(1,1)) are the central models in time series analysis. A Bayesian approach requires the user to define a prior distribution for the dependencies of these models. Understanding and interpretation of the priors is quite difficult in general, although it is very much desired to ensure that the priors behave according to the users prior knowledge about the process. In this report, we approach this problem using the recently developed ideas of the penalized complexity (PC) priors. These priors have important properties like robustness and invariance to reparameterisations, as well as a clear interpretation. A PC prior is computed based on the specific principles, where the model component complexity is penalized in terms of deviation from simple/base model formulations. In this report, the PC prior framework is applied to construct the prior distributions for dependencies of the AR(1) processes, the MA(1) processes and the ARMA(1,1) processes.

1 Introduction

In the Bayesian statistical inference of time series models, we assign prior distributions for all the hyper parameters of the model. The prior distribution about the hyper parameters represents our prior beliefs/understanding about the hyper parameter space. In general, it is very hard to express exact prior information about the hyper parameters. An expert knowledge is required to mention the concrete probabilistic information about the hyper parameters. More commonly the prior distributions used in general are not subjective and are open to criticism.

There are several reasons for using non-subjective priors ranging from the lack of expert information, to the difficulty in eliciting information about structural parameters that are further down the model hierarchy, such as precision or correlation parameters. As models grow more complex, the difficulty in specifying expert priors on the parameters increases. Martyn Plummer, the author of JAGS software for Bayesian inference [1] goes so far as to say

"[...] nobody can express an informative prior in terms of the precision[...]"

Apart from the fully subjective expert priors, there are three main methods of selecting priors. The method of prior selection furthest from expert elicitation priors are "objective" priors (Bernardo, 1979 [2]; Berger, 2006 [3]; Berger et al., 2009 [4]; Ghosh, 2011 [5]). These priors try to provide as little information as possible into the inference procedure. Objective priors strongly depend on the design and have philosophical issues amongst Bayesians; example discussion contributions to Berger, 2006 [3] and Goldstein, 2006 [6], but results can still be useful in practice.

Jeffreys' non-informative priors and their extension "reference priors" (Berger et al.,2009 [4]) are most common in the family of objective priors. These priors are typically improper, and require attention to ensure posteriors to be proper. If chosen carefully it leads to correct estimates as shown by Kamary, 2014 [7]. However, objective priors are model dependent and difficult to derive except for the simple cases. Further, it is highly sensitive to the likelihood changes. The entire prior must be recomputed for small changes in the likelihood, in order to ensure propriety. This does not suit well with the practice of "building block" approach type's statistical applications. In spite of shortcomings, the reference prior framework is the only complete framework for specifying the prior distributions.

Between subjective and objective priors there is a realm of "weakly informative" priors (Gelman, 2006 [8]; Gelman et al., 2008 [9]; Evans and Jang, 2011 [10]; Polson and Scott, 2012 [11]). These priors are constructed by having weak prior knowledge about the process which is generating data. It is rare to be completely ignorant about the process. The use of weak prior knowledge is sufficient to regularize the extreme inferences that can be obtained using maximum likelihood or non-informative priors.

There is a third approach to prior selection that is to select priors from the literature. In the best cases, the chosen prior was originally selected in a careful, problem independent manner for a similar problem to the one the statistician is solving. More commonly, these priors have been carefully chosen for the problem they were designed to solve and are inappropriate for the new application. Other priors in the literature have been selected for purely computational reasons.

Penalized Complexity priors (PC priors) (Simpson et al., 2016 [12]) belong to the realm of weakly informative priors, where users have some useful information about the process. The information in these priors is specified in terms of four underlying principles. These principles help to communicate the exact information that is encoded in the prior in order to make it interpretable. PC priors have a single parameter that the user must set, which controls the amount of flexibility that parameter can specify in the model. This parameter can be set using "weak" information. The second purpose of building these priors from a set of principles is to allow us to change these principles when needed. This gives the PC prior framework the advantage of flexibility without sacrificing its simple structure. PC priors are general enough to be used in realistically complex statistical models and are straightforward enough to be used by general practitioners. Using only weak information, PC priors represent a unified prior specification with a clear meaning and interpretation. The underlying principles are designed so that desirable properties follow automatically: invariance regarding reparameterisations, connection to Jeffreys' prior, support of Occam's razor principle, and empirical robustness to the choice of the flexibility parameter. The PC prior approach is not restricted to any specific computational method as it is a principled approach to the prior construction and therefore relevant to any application involving Bayesian analysis.

In this report, we will develop PC priors for time series models such as auto regressive process of order 1 i.e. AR(1), moving average process of order 1 i.e. MA(1) and auto regressive and moving average process of order (1,1) i.e. ARMA(1,1). PC priors for stationary auto regressive process have already been developed by Sørbye and Rue, 2016 [13], while PC priors for MA(1) and ARMA(1,1) are developed for the first time here.

To best present this report, it is divided into these sections: In Section 2, preliminaries such as definitions and notations related to the time series models are discussed. This gives us the basic idea about the processes that are dealt in this report. In Section 3, the fundamentals to develop the PC priors have been explained with an example. This will prepare us to use the principles to construct PC priors for the processes of our interest. In Section 4, PC priors are constructed for the dependency factor of the time series processes in the following order: AR(1), MA(1) and ARMA(1,1). In Section 5, based on the developed PC priors in section 4, simulated time series data is fit using the INLA framework. This section provides insight about how to use the PC priors in the INLA framework. Section 6, concludes the work done in this report with future recommendations.

2 Preliminaries

Basic building blocks of time series models consist of AR(1) process and MA(1) process and ARMA(1,1) process, these models are widely applied to model time-varying stochastic processes, for example within finance, biostatistics and natural sciences (Brockwell and Davis, 2002 [14] ; Chatfield, 2003[15]; Prado and West, 2010[16]).

2.1 AR(1) Process

Generally, an AR(1) process is defined by the equation 1:

$$x_t = \phi x_t - 1 + a_t, \tag{1}$$

where $a_t \sim \mathcal{N}(0, \kappa^{-1})$, for $t = 2, 3, \ldots, n$. x_1 is assumed to follow mean 0 and marginal precision $\tau = \kappa(1 - \phi^2)$. In the AR(1) process, dependency is governed by the factor ϕ. An AR process is a stationary process if roots of the characteristic polynomial lie inside the unit circle. In case of the AR(1) process the characteristic polynomial is represented by

$$z - \phi$$

We have to limit $|\phi| < 1$ for stationary AR(1) process. Figure 1 shows two random realizations of the stationary AR(1) process with correlation factor $\phi = 0.1$ and $\phi = 0.9$ mixed with noise with standard deviation (σ_e) of .1 respectively.

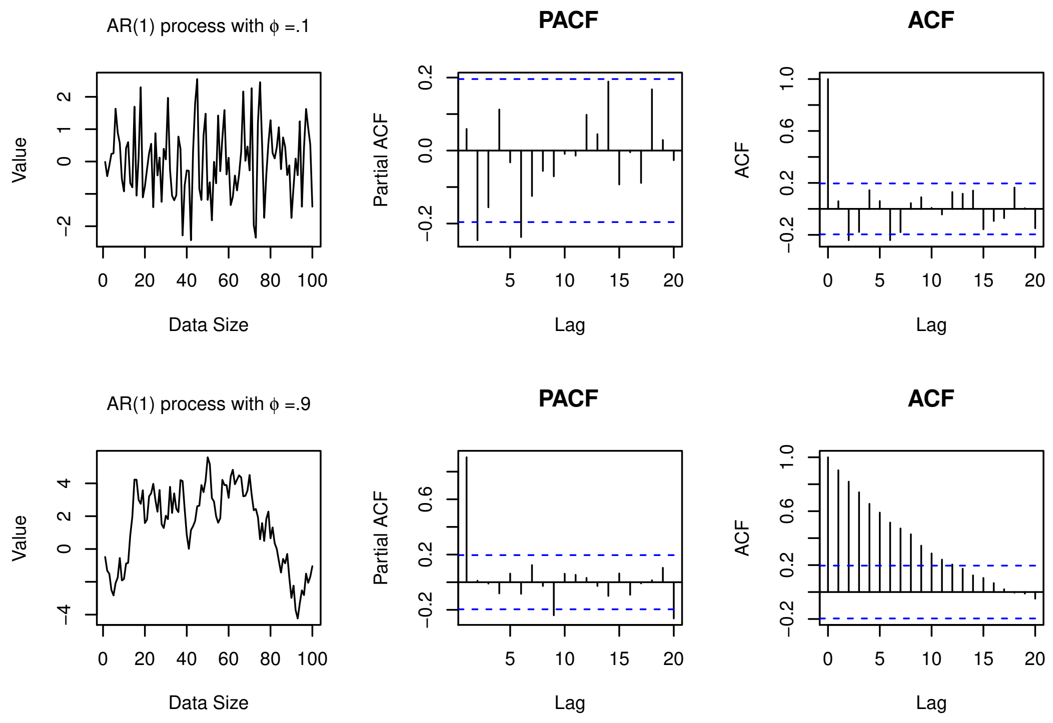

Figure 1: Random realizations of the AR(1) process with $\phi = .1$ and $\phi = .9$ mixed with noise with $\sigma_e = 0.1$

2.2 MA(1) Process

The MA(1) process is generally defined by the equation 2:

$$x_t = \theta a_{t-1} + a_t, \tag{2}$$

where $a_t \sim \mathcal{N}(0, \sigma^2)$, for $t = 2, 3, \ldots, n$. a_1 is assumed to be 0. In the MA(1) process dependency is governed by the factor θ. An MA process is called invertible MA process if it can be represented in term of AR series, for invertibility of the MA(1) process, constraint $|\theta| < 1$ must be followed. Figure 2 shows two random realizations of the invertible MA(1) process with correlation factor $\theta = 0.1$ and $\theta = 0.9$ mixed with noise with standard deviation(σ_e) of .1 respectively

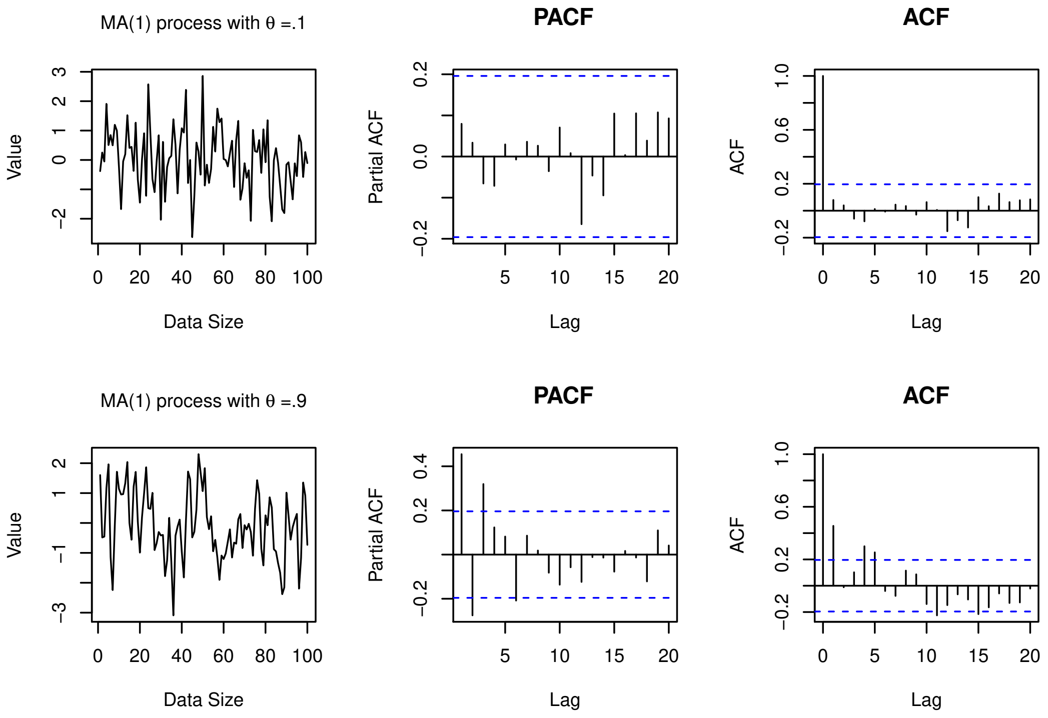

Figure 2: Random realizations of the MA(1) process with $\theta = .1$ and $\theta = .9$ mixed with noise with $\sigma_e = 0.1$

2.3 ARMA(1,1) Process

The ARMA(1,1) process is generally defined by the equation 3:

$$x_t = \phi x_{t-1} + \theta a_{t-1} + a_t, \tag{3}$$

where $a_t \sim \mathcal{N}(0, \sigma^2)$, for $t = 2, 3, \ldots, n$. a_1 is assumed to be 0. In an ARMA(1,1) process auto regressive dependency is governed by the factor ϕ and moving average part dependency is governed by θ, we have to limit $|\phi| < 1$ and $|\theta| < 1$ for stationary and invertible the ARMA(1,1). The ARMA(1,1) process with $|\phi| = 0$ is equivalent to MA(1) process and similarly, the ARMA(1,1) process with $|\theta| = 0$ is equivalent to AR(1) process.

Figure 3 represents random realizations of the ARMA(1,1) process with correlation factors $(\phi, \theta) = \{(.1, .1), (.4, .9), (.9, .4), (.9, .9)\}$ respectively mixed with noise with standard deviation (σ_e) of .1

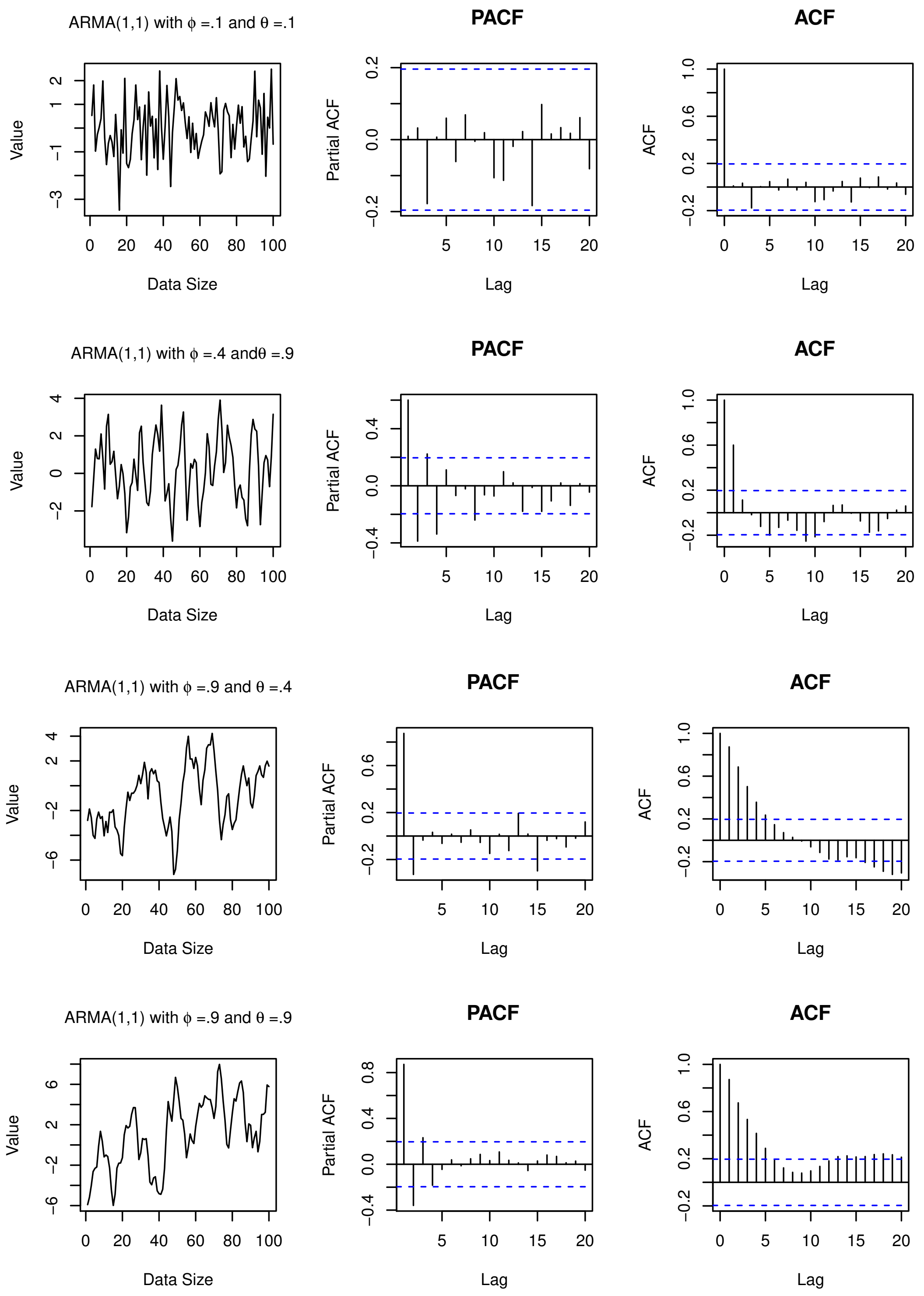

Figure 3: Random realizations of the ARMA(1,1) process with $(\phi, \theta) = (.1, .1)$, $(\phi, \theta) = (.4, .9)$, $(\phi, \theta) = (.9, .4)$, $(\phi, \theta) = (.9, .9)$ mixed with noise with $\sigma_e = 0.1$

3 Penalized Complexity (PC) Priors Framework

Penalized complexity priors [12] assign prior distributions for the hyper parameters of the model, based on the complexity of the model. As the name suggests, PC priors penalize priors for having more complex model. The basic idea for the construction of PC priors is that the priors for the base model (simpler model) are more probable then the priors for the complex models (flexible model). PC priors are constructed with the help of the following four principles:

3.1 Principle 1: Occam's Razor:

The principle of parsimony says that the simpler model formulation should be preferred until there is enough support for a more complex model. In this framework, the simpler model is the base model, so the priors will be penalized for deviating from the base model. This hints that as the complexity of the flexible model increases the priors for the flexible model will become less probable when compared to the priors for the base model, which imply that the prior densities of the hyper parameters should decay as the complexity of the flexible model increases. The complexity of the flexible model compared to the based model is also a measure of the distance between two models. But there arises the question how to measure the distance between two models or what are the measure of complexity in this framework. This introduces the next principle for measuring the complexity between two models.

3.2 Principle 2: Measure of Complexity:

The Kullback -Leibler divergence (KLD) is used to measure of the increased complexity between two probability distributions. Between two probability densities f and g, KLD is defined by equation 4:

$$KLD(f||g) = \int f(x) \log \frac{f(x)}{g(x)} dx \tag{4}$$

KLD is a measure of the information lost when the base model g is used to approximate the flexible model f. Note, that this is an asymmetric function, which means, the measure of complexity is non symmetric, hence the distance considered by the KLD is not a metric. Since, this measure of complexity is generated through integration of densities; it doesn't match with the notion of the distance dimensionally. Hence, the unidirectional measure of the distance $d(f||g) = \sqrt{(2KLD(f||g))}$ is used. d is considered to be measure of the distance based on the complexity of the model f when compared to the model g. The factor of "2" is chosen for the sake of convenience.

Now, a suitable measure of the distance between the base model and the flexible model, in the form of the KLD measure, is established.

It is known that the priors are to be penalized with reference to this measure of distance. But, the framework is still not prepared to construct the PC prior, as how much prior distribution need to be penalized quantitatively is not known as of now. This leads to the third principle.

3.3 Principle 3: Constant rate penalization:

While choosing the prior distribution for the distance measure d, it is natural to assume that the mode of the prior distribution should be located at the distance which corresponds to the base model i.e. $d = 0$, while the density decays as the distance from the base model increases, so that the prior densities for distance d must satisfy the equation 5:

$$\frac{\pi_d(d+\delta)}{\pi_d(d)} = r^{\delta}, d, \delta \geq 0 \tag{5}$$

for some constant r with $0 < r < 1$. The idea behind constant rate penalization is that the relative change in the prior densities doesn't depend on the distance d when the complexity of the flexible model increases from the distance d to the distance $d \pm \delta$. Since it is known that the distance $d = 0$ represent the base model, this idea of constant rate penalization assumption implies an exponential prior on the distance scale. So, it results in the prior distribution for the distance scale mention at equation 6

$$\begin{aligned} \pi_d(d) &= \lambda \exp(-\lambda d) \\ r &= \exp(-\lambda) \end{aligned} \tag{6}$$

It should be noted that the distance (d) between the base model and the flexible model is measured using the KLD, is a function of ξ (hyper parameters for the model). Let's say, the distance d is denoted as a function of ξ as $d(\xi)$. Now, from the constant rate penalization, it is known that the distance $d(\xi) \sim \exp(\lambda)$. So, by applying transformation of random variable to the prior distributions, the prior distribution for hyper parameters is given by equation 7:

$$\pi(\xi) = \pi(d(\xi)) | \frac{\partial d(\xi)}{\partial \xi} | \tag{7}$$

Now, the PC prior for our hyper parameter of interest ξ is constructed, the PC prior density is also a function of λ, which is unknown as of now. This introduces the next principle, for interpretation of λ.

3.4 Principle 4: User defined scaling:

This discussion was started by saying that the PC priors are the weakly informative priors, however till now no information about the hyper parameters space has been introduced, construction till now is generic and may be applied for the hyper parameters of any particular class of models, however in the real life situation, it would be applied to a particular problem or a particular model. For the particular problem or a model of interest, user must have a broad idea about the sensible upper bound U for the parameter of interest and α the tail event which is put about this sensible upper bound. The use of U, α also give a significance to the unknown parameter λ. This prior knowledge is used at equation 8 for finding out the unknown scaling factor λ for the constructed PC prior.

$$\Pr\left(Q(\xi) > U\right) = \alpha \tag{8}$$

Where $Q(\xi)$ is an interpret-able transformation of the flexibility parameter.

It should be noted that the idea of applying the PC prior is very useful, when the user has some vague idea about the the hyper parameters available. The vague idea can be in the following form,

> ...it is unlikely that parameter is larger than some number ...

Now, the framework based on these 4 principles is ready for construction of the PC priors, construction of the PC prior for the precision parameter (τ) for a multivariate normal distribution is an example before starting construction of the PC priors for the dependencies of the time series models.

Example : Construction of the PC Priors for the precision parameter (τ) in the multivariate Normal distribution:

Let $\mathcal{N}_0^p(\mu_0, \Sigma_0)$ denote the base model which follows a multivariate normal distribution with dimension p. And the flexible model is of the form $\mathcal{N}_1{}^p(\mu_1, \Sigma_1)$, using the KLD to calculate the measure of the distance between two models,

$$KLD(\mathcal{N}_1^p||\mathcal{N}_0^p) = \frac{1}{2}\{tr(\Sigma_0^{-1}\Sigma_1) + (\mu_0 - \mu_1)^T\Sigma_0^{-1}(\mu_0 - \mu_1) - p - \log\frac{|\Sigma_1|}{|\Sigma_0|}\} \tag{9}$$

The PC priors are formulated for the precision parameter (τ), so simplest assumption for the base model is that it is a model with no random effect and for the flexible model is to add random effects to the base model i.e. $\Sigma_0 = 0$, which is not useful directly for calculating the distance between two models. So, $\Sigma_0 = \frac{R}{\tau_0}$ and $\Sigma_1 = \frac{R}{\tau}$ is assumed and then the limit $\lim \tau_0 \to \infty$ is taken to make the base model with no random effect. R is assumed to be full rank fixed matrix. μ_0 and μ_1 are assumed to be 0 vectors. The KLD based distance between the base model and the flexible model is formulated at equation 11:

$$KLD = \frac{p}{2}\frac{\tau_0}{\tau}\{1 + \frac{\tau}{\tau_0}\log(\frac{\tau}{\tau_0}) - \frac{\tau}{\tau_0}\} \tag{10}$$

Now, as $\tau_0 \gg \tau$ i.e. equivalent to that when τ_0 goes to ∞, then $\frac{\tau}{\tau_0}, \frac{\tau}{\tau_0}\log(\frac{\tau}{\tau_0})$ will go to 0, which gives

$$d(\tau) = \sqrt{\frac{p\tau_0}{\tau}} \tag{11}$$

Now, the distance between two models in formulated in terms of the hyper parameter, principle 3 and principle 4 is applied to get the exact prior distribution for the precision parameter (τ) in this situation. Since, $d(\tau)$ is a function of τ and it is also known that the distance between two models follows exponential distribution with rate λ so by transforming the variables, the prior distribution for precision τ is constructed at equation 12 :

$$\begin{aligned} |\frac{\partial d(\tau)}{\partial \tau}| &= \frac{1}{2}\sqrt{\frac{p\tau_0}{\tau^3}} \\ \pi(d(\tau)) &= \lambda \exp -\lambda d(\tau) \\ \pi(\tau) &= \frac{\delta}{2}\tau^{\frac{-3}{2}} \exp\frac{-\delta}{\sqrt{\tau}} \end{aligned} \tag{12}$$

where $\delta = (-\lambda\sqrt{p\tau_0})$. This prior distribution is type -2 Gumbel distribution. To find the significance of the scaling parameter δ , principle 4. (i.e. probability statement) is applied

$$\Pr(\frac{1}{\sqrt{\tau}} > U) = \alpha$$

$$\delta = -\frac{\log\alpha}{U}$$

For the different values of the scaling parameter $\delta = c(.2, 1, 10)$ plots of the PC prior for the precision parameter (τ) are shown at the Figure 4,

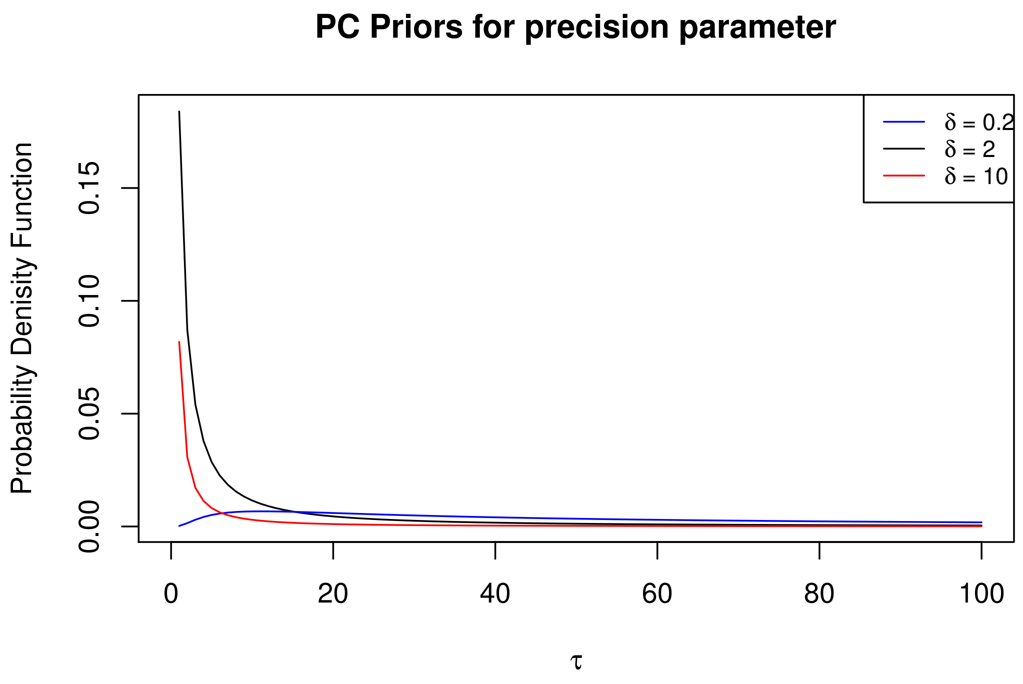

Figure 4: PC Prior for precision parameter (τ) in the multivariate normal distribution for different scaling parameters

4 Construction of the PC priors for the dependencies of time series models

As discussed in the Section 2, building blocks for the time series models are the AR(1) processes, the MA(1) processes and the ARMA(1,1) processes. In this section, the PC prior is developed for the dependency factor at lag one of an AR(1) process and an MA(1) process. In case of the ARMA(1,1) process, the joint prior distribution for the dependencies of AR part and of MA part is constructed. Key in constructing the PC priors lie in the way, the base model is selected. This will become clearer with the progress of the section.

4.1 Construction of the PC prior for the dependency at lag one of an AR(1) process

The general representation of an AR(1) process is defined by equation 1. In the case of an AR(1) process, there are two choices for the base model, i.e. $\phi = 0$ or $\phi = 1$. Depending on the particular problem user may choose any one of them. One can choose the base model to be independent with reference to time i.e. no dependency in time that corresponds to $\phi = 0$ for the base model. And one can also choose the base model to be no change with reference to to time that corresponds to the case when $\phi = 1$ for the base model. Construction of the PC priors for the dependency factor (ϕ) in each case will follow the same principles; however both approaches will lead to the different prior distributions for ϕ. For simplification of the calculations, the precision for the noise is assumed to be known and fixed.

4.1.1 Construction of the PC priors for the dependency at lag one of an AR(1) process when the base Model: No dependency in time

Let's say p is the dimension of the data-set, the base model is considered with the dependency parameter $\phi = 0$ and the flexible model is considered with the dependency parameter ϕ. For calculating the distance $d(\phi)$ based on the KLD, values of μ_0 and μ_1 and Σ_0 and Σ_1 are required. $\mu_0 = \mu_1 = 0$ and Σ_0 is an identity matrix of order p multiplied by a factor of $\frac{1}{\tau}$, and Σ_1 is such that $(\Sigma_{ij}) = \frac{1}{\tau}\phi^{|i-j|}$,

$$\Sigma_1 = \frac{1}{\tau}\begin{bmatrix} 1 & \phi & \phi^2 & \dots & \phi^{p-1} \\ \phi & 1 & \phi^2 & \dots & \phi^{p-2} \\ \phi^2 & \phi & 1 & \dots & \phi^{p-3} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \phi^{p-1} & \phi^{p-2} & \phi^{p-3} & \dots & 1 \end{bmatrix} \tag{13}$$

$${\Sigma_1}^{-1} = \frac{\tau}{(1-\phi^2)}\begin{bmatrix} 1 & -\phi & 0 & \cdots & \cdots & \cdots & \cdots & 0 \\ -\phi & (1+\phi^2) & -\phi & \ddots & & & & \vdots \\ 0 & -\phi & (1+\phi^2) & -\phi & \ddots & & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & & \vdots \\ \vdots & & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & & \ddots & -\phi & (1+\phi^2) & -\phi & 0 \\ \vdots & & & & \ddots & -\phi & (1+\phi^2) & -\phi \\ 0 & \cdots & \cdots & \cdots & \cdots & 0 & -\phi & 1 \end{bmatrix} \tag{14}$$

$$det(\Sigma_1) = \frac{1}{\tau^p}(1-\phi^2)^{p-1} \tag{15}$$

$$det(\Sigma_0) = \frac{1}{\tau^p} \tag{16}$$

Using the above information the KLD between the base model and the flexible model is formulated at equation 17 :

$$\begin{aligned} KLD(\mathcal{N}_1^p||\mathcal{N}_0^p) &= \frac{1}{2}\{tr(\Sigma_0^{-1}\Sigma_1) + (\mu_0-\mu_1)^T\Sigma_0^{-1}(\mu_0-\mu_1) - p - \log\frac{|\Sigma_1|}{|\Sigma_0|}\} \\ KLD(\mathcal{N}_1^p||\mathcal{N}_0^p) &= \frac{1}{2}(1-p)\log(1-\phi^2) \\ d(\phi) &= \sqrt{(1-p)\log(1-\phi^2)} \end{aligned} \tag{17}$$

Since, the distance is formulated as a function of ϕ, using the principle 3, the prior distribution for the ϕ is constructed at equation 19:

$$\begin{aligned} |\frac{\partial d(\phi)}{\partial \phi}| &= \frac{1}{2}\sqrt{\frac{1-p}{\log(1-\phi^2)}}\frac{2|\phi|}{(1-\phi^2)} \\ \pi(d(\phi)) &= \lambda\exp(-\lambda d(\phi)) \end{aligned} \tag{18}$$

$$\pi(\phi) = \delta\exp(-\delta\sqrt{-\log(1-\phi^2)})\frac{\phi}{(1-\phi^2)\sqrt{-\log(1-\phi^2)}} \tag{19}$$

Where the scaling parameter $\delta = \lambda\sqrt{p-1}$. To find significance of the scaling parameter δ, principle 4 is applied. The probability statement is defined by $\Pr(|\phi| > U) = \alpha$ which gives significance to the scaling parameter at equation 20:

$$\delta = \frac{-\log \alpha}{\sqrt{-\log\left(1-U^2\right)}} \tag{20}$$

Plots of the PC priors for correlation at lag one are shown by the Figure 5, we have set three different values of the scaling paramter $\delta = c(100, 10, 1)$

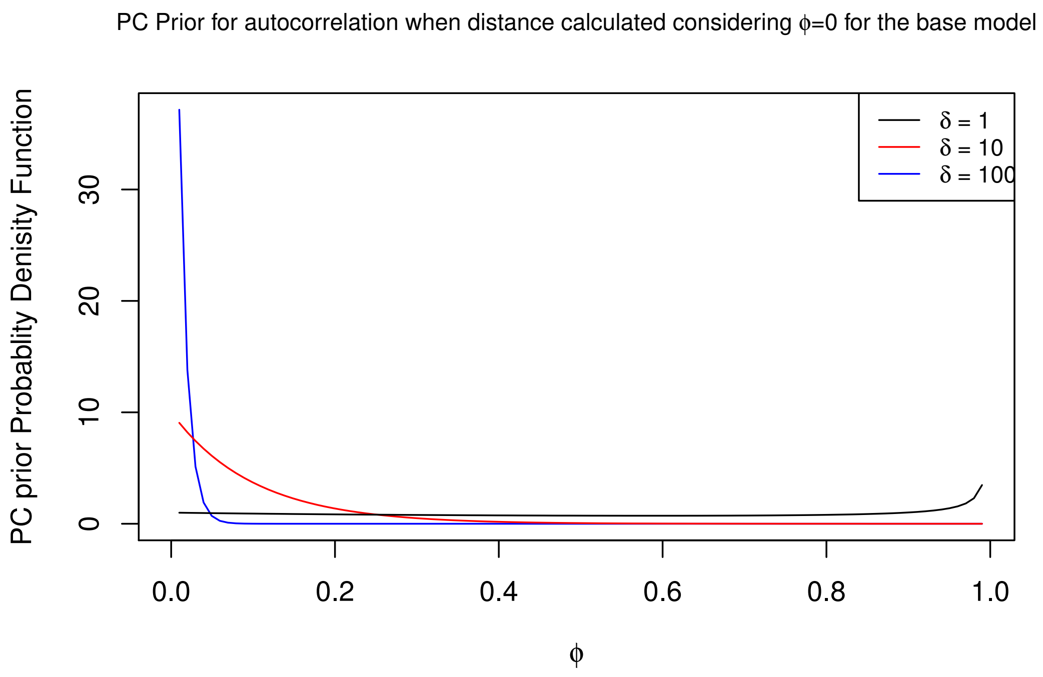

Figure 5: PC Prior for correlation at lag 1 (ϕ), of an AR(1) process with different scaling parameters

It is observed that for different values of the scaling parameter δ, the PC prior behaves both like an informative prior and an uninformative prior. From the Figure 5, it is observed that when the scaling parameter δ is set to 100, then, the PC prior shows shrinkage and becomes informative prior whereas in the case when the scaling parameter was set to 1 it becomes a flat prior.

4.1.2 Construction of the PC priors for the dependency at lag one of an AR(1) process when the Base Model: No change in time

An alternative approach for considering the base model can be such that the base model does not change with time ($\phi = 1$). This represents the limiting case of random walk, which is a non stationary and a singular process. So dependencies for the base model and the for flexible model are assumed to be ϕ_0 and ϕ, such that $\phi_0 > \phi$. And the limiting case will be discussed when the limit $\lim \phi_0 \to 1$. In this situation, Σ_0 and Σ_1 for our base and flexible models are given by equation 21:

$$\Sigma_0 = \frac{1}{\tau}\begin{bmatrix} 1 & \phi_0 & \phi_0^2 & \dots & \phi_0^{p-1} \\ \phi_0 & 1 & \phi_0^2 & \dots & \phi_0^{p-2} \\ \phi_0^2 & \phi_0 & 1 & \dots & \phi_0^{p-3} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \phi_0^{p-1} & \phi_0^{p-2} & \phi_0^{p-3} & \dots & 1 \end{bmatrix} \quad \Sigma_1 = \frac{1}{\tau}\begin{bmatrix} 1 & \phi & \phi^2 & \dots & \phi^{p-1} \\ \phi & 1 & \phi^2 & \dots & \phi^{p-2} \\ \phi^2 & \phi & 1 & \dots & \phi^{p-3} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \phi^{p-1} & \phi^{p-2} & \phi^{p-3} & \dots & 1 \end{bmatrix} \tag{21}$$

This gives

$$KLD(\mathcal{N}_1^p||\mathcal{N}_0^p) = \frac{1}{2}\Big[\frac{1}{1-\phi_0^2}\{p - 2(p-1)\phi_0\phi + (p-2)\phi_0^2\} - p - (p-1)\log\frac{(1-\phi^2)}{(1-\phi_0^2)}\Big] \tag{22}$$

While considering the limiting case $\lim \phi_0 \to 1$, we have

$$d(\phi) = \sqrt{2KLD} = \sqrt{\frac{2(p-1)(1-\phi)}{1-\phi_0^2}} = c\sqrt{1-\phi} \tag{23}$$

where $|\phi| < 1$ and c is independent from ϕ. It should also be noted that $d(\phi) \leq c\sqrt{2}$. Since, $d(\phi)$ now has a range so we have to use truncated exponential distribution. In this case, the prior for ϕ is constructed at equation 27:

$$d(\phi) = c\sqrt{1-\phi} \tag{24}$$

$$\pi(\phi) = \frac{\lambda \exp(-\lambda d(\phi))}{1-\exp(-\sqrt{2}\lambda c)} \tag{25}$$

$$\left|\frac{\partial d(\phi)}{\partial \phi}\right| = \frac{c}{2\sqrt{1-\phi}} \tag{26}$$

$$\pi(\phi) = \frac{1}{2}\frac{\delta \exp(-\delta\sqrt{1-\phi})}{(1-\exp(-\sqrt{2}\delta))\sqrt{1-\phi}} \tag{27}$$

Where $\delta = \lambda c$. However, we have to use principle 4 for the significance of the scaling parameter δ, the probability statement $\Pr(|\phi| > U) = \alpha$ is defined to give significance to the scaling parameter δ at equation 28:

$$\alpha = \frac{1-\exp(-\delta\sqrt{1-U})}{(1-\exp(-\sqrt{2}\delta))} \tag{28}$$

We must note:

$$\alpha_{min} = \sqrt{\frac{1-U}{2}} \tag{29}$$

Plots of the PC priors for correlation at lag one are shown at the Figure 6, we have set three different values of the scaling parameter $\delta = c(.1, 1, 5)$ It is observed that for the different values of the scaling parameter δ, the PC prior behaves both like an informative prior and an uninformative prior. It is also noticed that increasing values of the scaling parameter δ increases the amount of information about the data, which corresponds to more shrinkage of the prior towards the base model. However, by changing the user defined scale, PC prior can easily constructed as an uninformative/flat prior also.

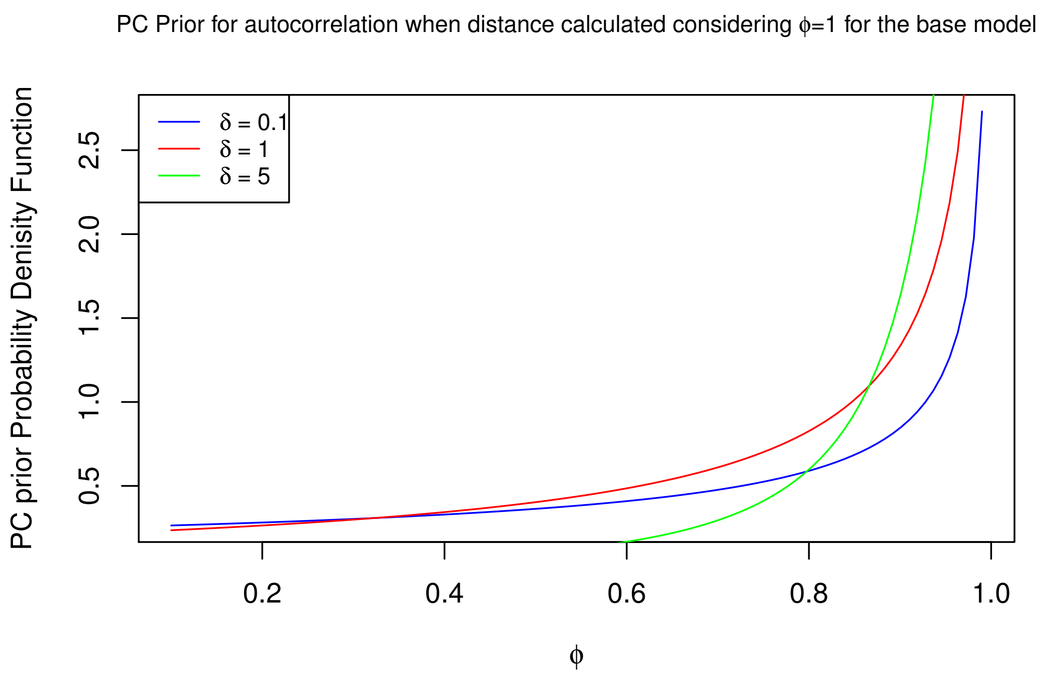

Figure 6: PC Prior for correlation at lag 1 (ϕ), of an AR(1) process with different scaling parameters

4.1.3 Comparison of the PC Priors of AR(1) process with the Reference and the Jeffery's Priors

In this section, the PC priors developed for correlation of an AR(1) process are compared with the existing priors. In Bayesian setting, some time priors are preferred that doesn't strongly influence the posterior distributions. These priors are called uninformative priors. We can assign uniform prior to our hyper parameter but that may not be invariant towards reparameterization. So the Jeffery's priors are the generalization of the idea of uninformative priors. And the Jeffery's priors are based on the principle of invariance. Jeffery's prior are calculated at equation 30 - equation 33 [19]:

$$\pi_J(\theta) = \sqrt{det(I(\theta))} \tag{30}$$

where $I_{ij}(\theta)$ = Fisher Information matrix = $-E_\theta(\frac{\partial^2 \log{(pX|\theta)}}{\partial\theta_i \partial\theta_j})$, in the case of AR(1) process,

$$det(I(\theta)) = (\{\frac{n}{1-\phi^2} + \frac{1-\phi^{2n}}{1-\phi^2}\{E(\frac{X_0^2}{\sigma^2}) + \frac{1}{1-\phi^2}\}\}) \tag{31}$$

which gives us

$$\pi(\phi) \propto \frac{(1-\phi)^2}{\sigma^2}\sqrt{\frac{1}{1-\phi^2}(n + \frac{1-\phi^{2n}}{1-\phi^2})} \tag{32}$$

which is an un-normalized density, which simplifies for large values of n:

$$\pi(\phi) \propto \sqrt{\frac{1-\phi}{1+\phi}}(1-\phi) \tag{33}$$

It is desired to construct prior distribution for the notion of the distance using the Jeffery's prior distribution. We have two notions of the distance depending on the base model. In the first case when the base model with $\phi = 0$ is considered, resulted $d(\phi) = \sqrt{(1-p)\log(1-\phi^2)}$ which gives

$$\phi = \sqrt{1 - \exp(-\frac{d^2}{n-1})} \tag{34}$$

prior distribution for this notion of distance is formulated at equation 36:

$$\pi_J(d) = \pi_J(\phi)|\frac{\partial(\phi)}{\partial d}| \tag{35}$$

which gives

$$\pi_J(d) \propto \frac{\exp\frac{-d^2}{2(n-1)}(1 - \sqrt{(1-\exp\frac{-d^2}{(n-1)})})^2(d)}{\sqrt{(1-\exp\frac{-d^2}{(n-1)})}} \tag{36}$$

Plot of the prior distribution for the notion of distance is given by the Figure 7:

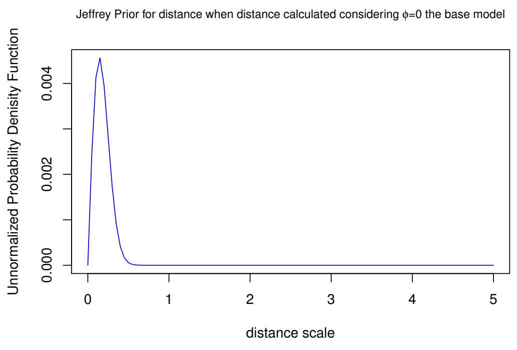

Figure 7: Jeffrey's prior for the notion of distance

Now, the prior distribution is constructed for the case when base model is assumed with $\phi = 1$, $d(\phi) = c\sqrt{1-\phi}$. which gives us

$$\phi = 1 + \frac{-d^2}{c} \tag{37}$$

for the sake of simplicity, assume $c = 1$, which will put a constraints on d that $d \leq 1$. The prior distribution is formulated for notion of distance at equation 39:

$$\pi_J(d) = \pi_J(\phi)|\frac{\partial(\phi)}{\partial d}| \tag{38}$$

which gives

$$\pi_J(d) \propto \frac{d^4}{\sqrt{(2-d^2)}} \tag{39}$$

Plot of the prior for the notion of the distance is given by the Figure 8:

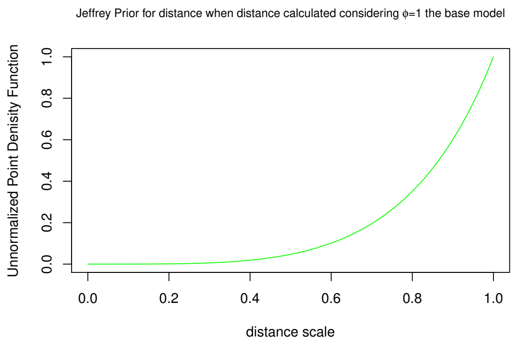

Figure 8: Jeffrey's prior for the notion of distance

It is observed that for both choices of the base models, prior distribution constructed for the notion of the distance considering Jeffreys prior distribution for the parameter overfits. It gives zero probability mass for the cases when d is close to 0, which contradicts the assumptions of the Occam's Razor. It is observed that the prior distribution makes some distances which are not close to the base model more probable. Jeffrey priors are often used for defining the prior distributions for the dependency factor of an AR(1) process. It should be considered that origin of the Jeffreys prior is to give uninformative priors which are invariant under reparameterization. However, if they are looked with the perspective of the distance from the base model, they don't support the idea that the base model is more probable.

Now, prior distribution for the notion of distance is constructed considering the reference prior distribution for the parameter (ϕ) of the model. The idea behind the reference priors is to formalize what exactly meant by an "uninformative prior". It is a function that maximizes some measure of distance or divergence between the posterior and prior. Reference Prior for correlation at lag 1 of an AR(1) process are given at equation 40 [20]:

$$\pi_R(\phi) = \frac{1}{\pi}\frac{1}{\sqrt{(1-\phi^2)}} \tag{40}$$

There are two notions of the distance available depending on the base model. In the first case when the base model is considered with $\phi = 0$, $d(\phi) = \sqrt{(1-p)\log(1-\phi^2)}$ which leads to

$$\phi = \sqrt{1 - \exp(-\frac{d^2}{n-1})} \tag{41}$$

So prior distribution for the notion of distance is formulated at equation 43:

$$\pi_R(d) = \pi_R(\phi)|\frac{\partial(\phi)}{\partial d}| \tag{42}$$

Which gives

$$\pi_R(d) = \frac{1}{\pi}\frac{\exp(-\frac{d^2}{2(n-1)})(1-\exp-\frac{d^2}{n-1})^{-.5}d}{(n-1)} \tag{43}$$

Plot of the prior distribution for this notion of distance is given by the Figure 9:

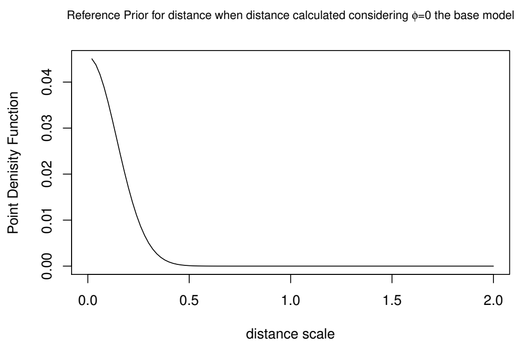

Figure 9: reference prior for the notion of distance

Now, prior distribution is constructed for the case when base model is assumed with $\phi = 1$, $d(\phi) = c\sqrt{1-\phi}$. which gives

$$\phi = 1 + \frac{-d^2}{c} \tag{44}$$

for the sake of simplicity we assume that c =1. Which will put a constraints on d that $d \leq 1$. The prior distribution for this notion of distance is formulated at equation 46:

$$\pi_R(d) = \pi_R(\phi) | \frac{\partial(\phi)}{\partial d} | \tag{45}$$

which gives

$$\pi_R(d) = \frac{2}{\pi} \frac{1}{\sqrt{2-d^2}} \tag{46}$$

Plot of the prior distribution for this notion of distance is given by Figure 10:

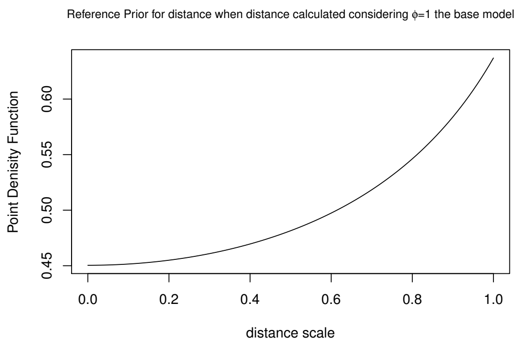

Figure 10: reference prior for the notion of distance

The reference prior looks reasonable when we consider the distance from the base model corresponding to $\phi = 0$, as it doesn't overfit distances close to 0. It supports the idea that models close to the base model are more probable than the model far from the base model. However, if the base model is chosen with $\phi = 1$, reference prior doesn't looks that reasonable as it makes probable those models which are far from the base model. Clearly, when the ϕ is close to 0, reference prior may be used. But ϕ is close to 1 for a particular problem, the reference prior approach may not be suitable. PC Priors are based on constant rate penalization principle hence they support both base models and can be used for any general or specific setting of ϕ.

4.2 Construction of the PC priors for the dependency at lag one of an MA(1) process

The general representation of MA(1) process is defined by equation 2. In this section, the objective is to construct prior distribution for the parameter θ. There are two choices for the base model: $\theta = 0$ or $\theta = 1$. However, choosing the base model to be independent with reference to time i.e. no dependency in time which corresponds to $\theta = 0$ is more natural and intuitive for the MA processes. Choosing the base model to be no change with reference to to time which corresponds to the case when $\theta = 1$, but it is counter intuitive. This can be understood by the fact that measurement of the statistical signal is done assuming initial error to be 0, and the MA(1) process is statistically modelled using the difference of signals at consecutive time unit. It is natural and simpler to believe that the base model will have no dependency with time, and by observing the data it is sensed that there is a dependency among the signals in terms of the correlation factor (θ).

Let's say p is the dimension of the data-set, $\theta = 0$ is considered for the base model and θ for the flexible model. For calculating the KLD and the distance $d(\theta)$ between the base model and the flexible model, values of μ_0 and μ_1 and Σ_0 and Σ_1 are required. $\mu_0 = \mu_1 = 0$ and Σ_0 is an identity matrix of order p multiplied by a factor of σ^2, and Σ_1 is represented by equation 47:

$$\Sigma_{ij} = \left\{ \begin{array}{ll} \sigma^2(1+\theta^2) & |i-j| = 0, \\ \sigma^2(\theta) & |i-j| = 1, \\ 0 & \text{otherwise} \end{array} \right\} \tag{47}$$

$$\Sigma_1 = \sigma^2 \begin{bmatrix} (1+\theta^2) & \theta & 0 & \cdots & \cdots & \cdots & \cdots & 0 \\ \theta & (1+\theta^2) & \theta & \ddots & & & & \vdots \\ 0 & \theta & (1+\theta^2) & \theta & \ddots & & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & & \vdots \\ \vdots & & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & & \ddots & \theta & (1+\theta^2) & \theta & 0 \\ \vdots & & & & \ddots & \theta & (1+\theta^2) & \theta \\ 0 & \cdots & \cdots & \cdots & \cdots & 0 & \theta & (1+\theta^2) \end{bmatrix} \tag{48}$$

It is quite difficult to formulate the divergence between the the flexible model and the base model using the KLD by the given Σ_1, as it requires determinant of Σ_1 to be formulated in the closed form, however the determinant of Σ_1 doesn't have a closed form and it depends on the dimension of the Σ_1, numerical methods are required to find the determinant of Σ_1.

But after a keen observation of Σ_1, an interesting structure is found in Σ_1. It has the same structure as the inverse matrix shown at the equation 14. The only structural difference between the inverse matrix referred and the Σ_1 is the first and last entry of the matrix. The first and last entry in the matrix Σ_1 is $(1 + \theta^2)$, and the referred matrix has those entries as 1. The determinant of the referred matrix is available in the closed form.

This gives us the idea to approximate Σ_1 by Σ_{approx} by changing $\Sigma_{1_{(1,1)}}$ and $\Sigma_{1_{(p,p)}}$ to 1 instead of $(1+\theta^2)$, this approximation will result into a closed form of the determinant of Σ_{approx}. Σ_{approx}, its inverse and its determinant are represented at equation 49, equation 50 and equation 51:

$$\Sigma_{approx} = \sigma^2 \begin{bmatrix} 1 & \theta & 0 & \cdots & \cdots & \cdots & \cdots & 0 \\ \theta & (1+\theta^2) & \theta & \ddots & & & & \vdots \\ 0 & \theta & (1+\theta^2) & \theta & \ddots & & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & & \vdots \\ \vdots & & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & & \ddots & \theta & (1+\theta^2) & \theta & 0 \\ \vdots & & & & \ddots & \theta & (1+\theta^2) & \theta \\ 0 & \cdots & \cdots & \cdots & \cdots & 0 & \theta & 1 \end{bmatrix} \tag{49}$$

$$\Sigma_{approx}{}^{-1} = \frac{1}{\sigma^2(1-\theta^2)} \begin{bmatrix} 1 & (-\theta) & (-\theta)^2 & \dots & (-\theta)^{p-1} \\ -\theta & 1 & (-\theta)^2 & \dots & (-\theta)^{p-2} \\ (-\theta)^2 & (-\theta) & 1 & \dots & (-\theta)^{p-3} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ (-\theta)^{p-1} & (-\theta)^{p-2} & (-\theta)^{p-3} & \dots & 1 \end{bmatrix} \tag{50}$$

$$det(\Sigma_{approx}) = (\sigma^2)^p(1-\theta^2) \tag{51}$$

However, the approximation needs to be verified with reference to the measure of distance between the base model and the flexible model. The comparison of the actual distance and the approximated distance between the base model and the flexible model is studied by R Code at Appendix A.1. The actual distance is calculated using Σ_1 and approximated distance has been calculated using Σ_{approx}.

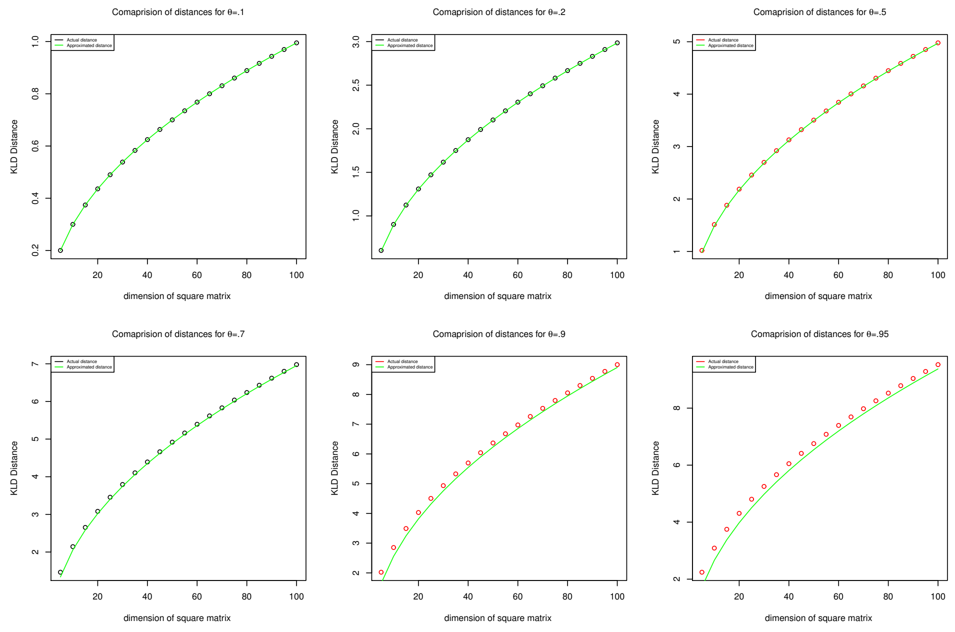

Figure 11: Comparison between the actual distance (considering Σ_1) with the approximated distance (considering Σ_{approx}) for different values of θ

It is observed from the Figure 11 that the approximation works well with reference to the KLD measure of the distance for almost all values of θ except very high values of θ. Hence, the approximation proposed is used for constructing the PC prior in this case:

$$KLD(\mathcal{N}_1^p||\mathcal{N}_0^p) = \frac{1}{2}\{tr(\Sigma_0^{-1}\Sigma_1) + (\mu_0 - \mu_1)^T\Sigma_0^{-1}(\mu_0 - \mu_1) - p - \log\frac{|\Sigma_{approx}|}{|\Sigma_0|}\} \tag{52}$$

$$KLD(\mathcal{N}_1^p||\mathcal{N}_0^p) = \frac{1}{2}((p-2)(\theta^2) - \log(1-\theta^2)) \tag{53}$$

$$d(\theta) = \sqrt{((p-2)(\theta^2) - \log(1-\theta^2))} \tag{54}$$

This can be further simplified using the approximation $\log(1-\theta^2) \approx -\theta^2$ when $|\theta| < 1$:

$$d(\theta) = \sqrt{((p-1)(\theta^2))} \tag{55}$$

This gives distance as a function of θ and the prior distribution for the θ is formulated at equation 57:

$$\pi(d(\theta)) = \lambda\exp(-\lambda d(\theta)) \tag{56}$$

$$\pi(\theta) = \delta\exp(-\delta|\theta|) \tag{57}$$

where $\delta = \lambda\sqrt{p-1}$. Since, there is a condition that $|\theta| < 1$ for invertible MA (1) process, the prior distribution formulated at equation 57 needs to be truncated.Truncated prior distribution is shown at equation 58

$$\pi(\theta) = \frac{\delta}{2(1-\exp(-\delta))}\exp(-\delta|\theta|) \tag{58}$$

To find significance of δ, the probability statement is defined:

$$\Pr(|\theta| > U) = \alpha$$

which needs to be solved with the help of numerical methods for δ. It is also be represented at equation 59

$$\alpha\exp(-\delta) + \exp(-U\delta) - \alpha = 0 \tag{59}$$

Uniroot command has been used to find the solution of this equation in the R-generic code for MA(1) process.

Plots of the PC prior for the correlation at lag one for the MA(1) process are shown at the Figure 12 for three different values of the scaling paramter δ i.e. $30, 10, 2$.

For the different values of the scaling parameter δ, the PC prior for θ behaves both like an informative prior and also like an uninformative/flat prior. In the Figure 12, for $\delta = 30$, the PC prior shows shrinkage and becomes informative prior whereas in the case when δ is set to 2, the PC prior behaves like a flat prior. It is also observed that as δ increases, the PC prior becomes more and more informative resulting into more shrinkage towards the base model.

4.3 Construction of the joint PC priors for the dependencies of the ARMA(1,1) process

The general representation of an ARMA(1,1) process is defined by equation 3. In this section, the objective is to construct PC prior distributions for the parameters θ and ϕ. There are three methods to approach this problem, the first method tries to construct the PC priors assuming the base model to be i.i.d., the second method formulates the PC priors, assuming the base model to be an AR(1) process and third method formulates the PC prior assuming the base model to be an MA(1) process. To formulate the joint PC priors for the ARMA(1,1) process in the first case, the base model is defined with $\theta = 0$ and $\phi = 0$. In the second case, the base model is defined to be an AR(1) process with ϕ. In the third case, the base model is defined to be MA(1) process with θ.

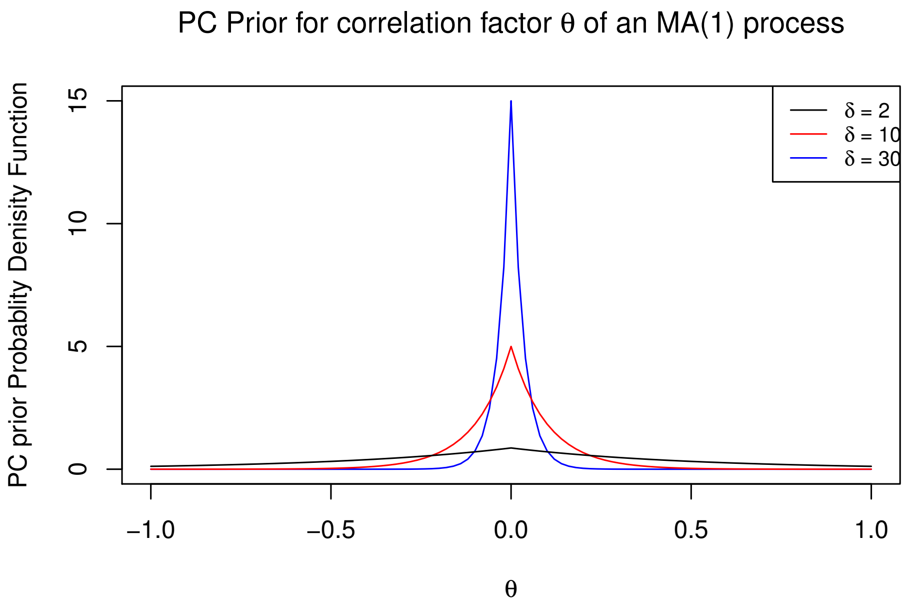

Figure 12: PC Prior for the correlation (θ) at lag 1, for MA(1) process for different scaling parameters

4.3.1 Construction of the joint PC priors for the dependencies of the ARMA(1,1) process for the base model to be iid

Let's say p is the dimension of the data-set, $(\phi = 0, \theta = 0)$ is considered for the base model and parameters (ϕ, θ) represents dependencies of the flexible ARMA(1,1) model. For calculating the distance $d(\phi, \theta)$ based on the KLD, values of μ_0 and μ_{ARMA} and Σ_0 and Σ_{ARMA} are required. $\mu_0 = \mu_{ARMA} = 0$ and Σ_0 is an identity matrix of order p multiplied by a factor of σ^2, and Σ_{ARMA} is represented by equation 60 and equation 61:

$$\Sigma_{ij} = \left\{ \begin{array}{ll} \gamma_0 & |i-j| = 0, \\ \gamma_1 & |i-j| = 1, \\ \phi^{|i-j|-1}\gamma_1 & otherwise \end{array} \right\} \tag{60}$$

where
$\gamma_0 = \sigma^2 \frac{(1+\theta^2+2\theta\phi)}{(1-\phi^2)}$
$\gamma_1 = \sigma^2 \frac{(1+\theta\phi)(\phi+\theta)}{(1-\phi^2)}$

$$\Sigma_{ARMA} = \begin{bmatrix} \gamma_0 & \gamma_1 & \phi^1\gamma_1 & \dots & \phi^{p-2}\gamma_1 \\ \gamma_1 & \gamma_0 & \gamma_1 & \dots & \phi^{p-3}\gamma_1 \\ \phi^1\gamma_1 & \gamma_1 & \gamma_0 & \dots & \phi^{p-3}\gamma_1 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \phi^{p-2}\gamma_1 & \phi^{p-2}\gamma_1 & \phi^{p-3}\gamma_1 & \dots & \gamma_0 \end{bmatrix} \tag{61}$$

It is quite difficult to formulate the the distance based on the KLD measure between the base model and the flexible model analytically using Σ_{ARMA}, as it requires the determinant of Σ_{ARMA} to be formulated in the closed form, however the determinant of Σ_{ARMA} doesn't have a closed form, the determinant of Σ_{ARMA} depends on the dimension of the matrix, and numerical methods are required to find the determinant of Σ_{ARMA}. But Σ_{ARMA} posses an interesting structure in it. It has the pattern of the co-variance matrix of the AR(1) process along with the subtle structural attributes similar to the co-variance matrix of the MA(1) process also. But it is dense unlike the co-variance matrix of an MA(1) processes because of the AR part in the ARMA(1,1) process. And unlike the AR(1) processes, it has a dense precision matrix because of the MA part in the ARMA(1,1) process.

A suitable, approximation of Σ_{ARMA} needs to be found in order to formulate its determinant in the closed form. While searching for a suitable approximation, the idea of approximation $\Sigma_{ARMA} \approx \Sigma_{AR}\Sigma_{MA}$ [17] came across. This approximation might lead to a smooth solution to the problem of finding determinant in the closed form. The determinant of Σ_{AR} has a closed form, whereas determinant of Σ_{MA} was formulated in the closed at equation 51. However, the approximation $\Sigma_{ARMA} \approx \Sigma_{AR}\Sigma_{MA}$ needs to be verified with reference to the measure of the distance between the base model and the flexible model. The comparison of the actual distance and the approximated distance between the base model and the flexible model is studied by R Code at Appendix A.2. The approximated distance, between the base model and the flexible model is calculated using the approximation that $\Sigma_{ARMA} \approx \Sigma_{AR}\Sigma_{MA}$.

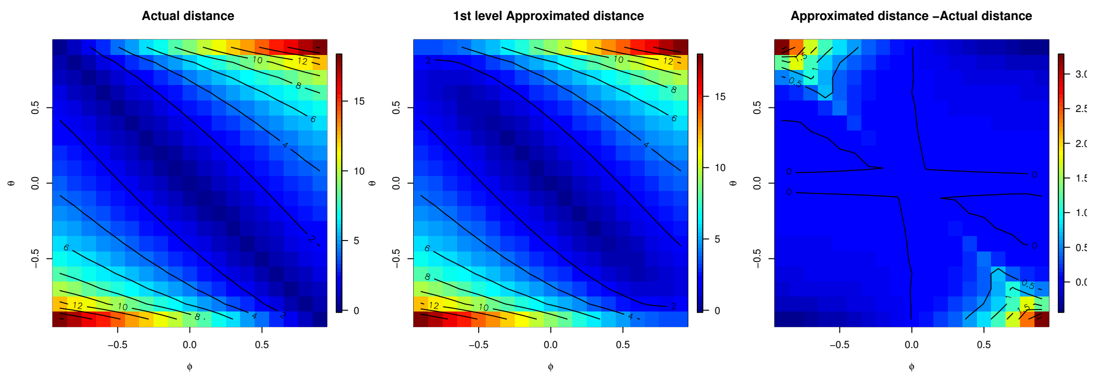

Figure 13: Comparison between the actual distance (considering Σ_{ARMA}) with the approximated distance (considering $\Sigma_{AR}\Sigma_{MA}$) when $n = 10$

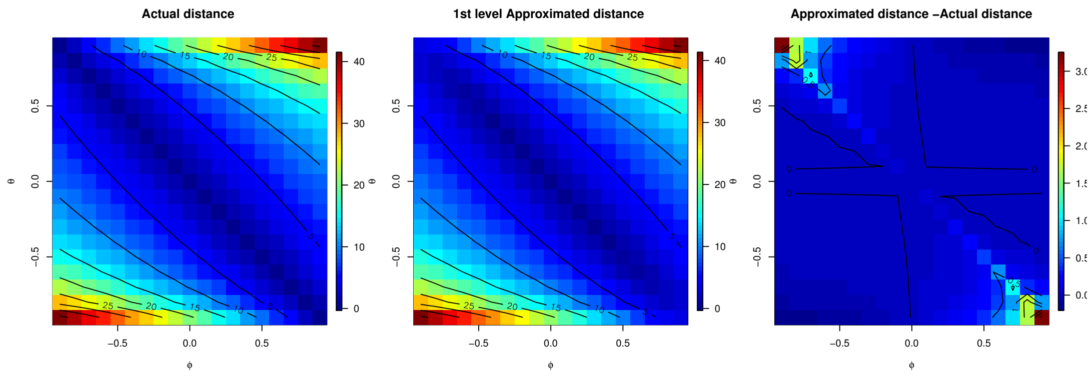

Figure 14: Comparison between the actual distance (considering Σ_{ARMA}) with the approximated distance (considering $\Sigma_{AR}\Sigma_{MA}$) when $n = 50$

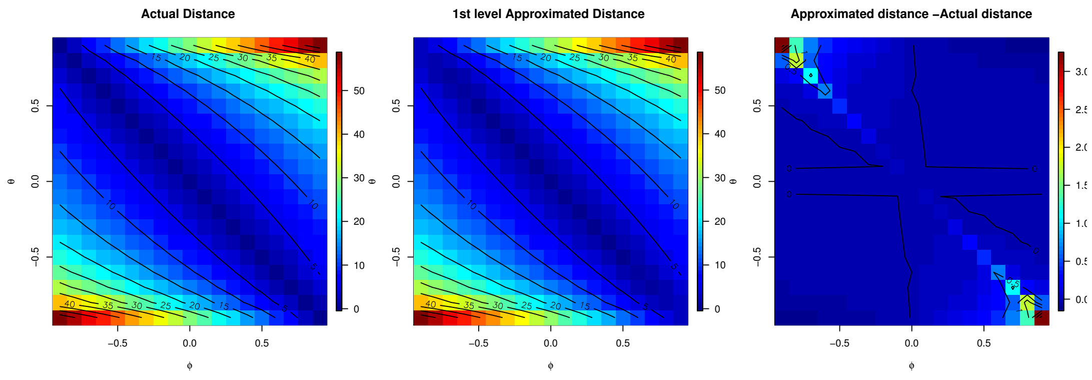

Figure 15: Comparison between the actual distance (considering Σ_{ARMA}) with the approximated distance (considering $\Sigma_{AR}\Sigma_{MA}$) when $n = 100$

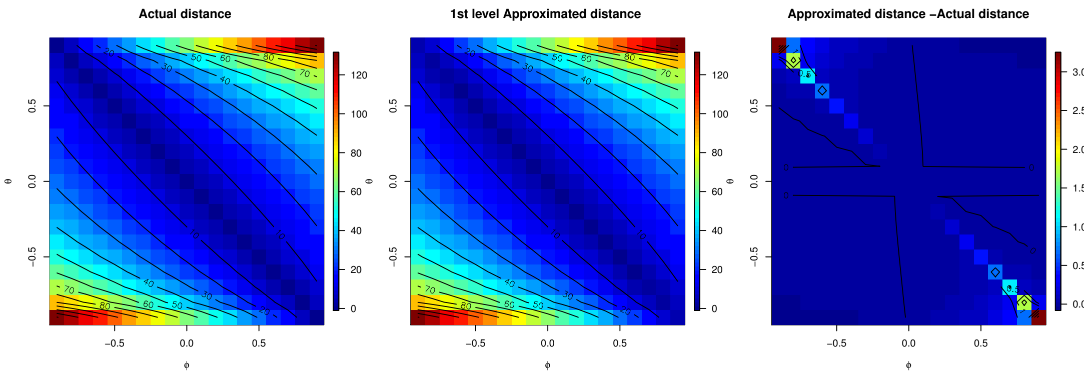

Figure 16: Comparison between the actual distance (considering Σ_{ARMA}) with the approximated distance (considering $\Sigma_{AR}\Sigma_{MA}$) when $n = 500$

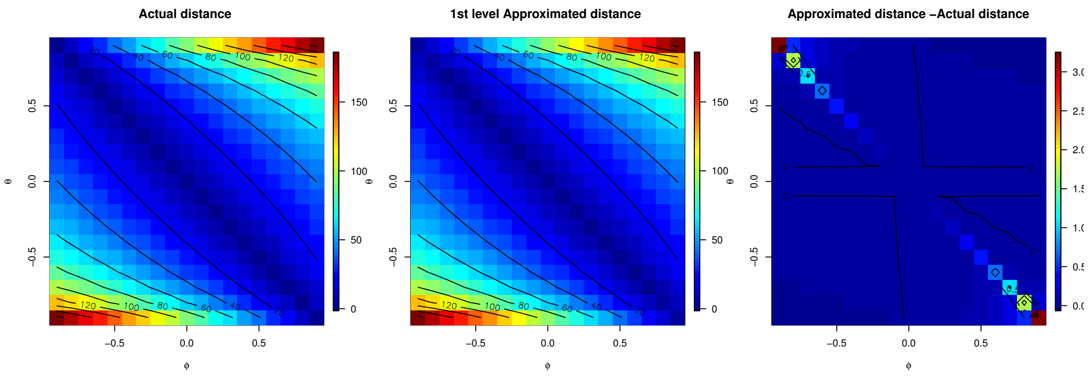

Figure 17: Comparison between the actual distance (considering Σ_{ARMA}) with the approximated distance (considering $\Sigma_{AR}\Sigma_{MA}$) when $n = 1000$

It is observed from the Figure 13 to the Figure 17 that the approximation ($\Sigma_{ARMA} \approx \Sigma_{AR}\Sigma_{MA}$) is a good approximation, with the increasing number of points in the data set the approximation improves significantly. This is observed by looking at the contour plots of difference between approximated distance and actual distance. Hence, the approximation proposed is used for constructing the PC prior in this case:

$$KLD(\mathcal{N}_1^p||\mathcal{N}_0^p) = \frac{1}{2}\{tr(\Sigma_0^{-1}\Sigma_1) + (\mu_0 - \mu_1)^T\Sigma_0^{-1}(\mu_0 - \mu_1) - p - \log\frac{|\Sigma_1|}{|\Sigma_0|}\} \tag{62}$$

$$KLD = \frac{1}{2}\{tr(\Sigma_0^{-1}\Sigma_{ARMA}) + -p - \log\frac{|\Sigma_{ARMA}|}{|\Sigma_0|}\} \tag{63}$$

$$KLD = \frac{1}{2}\{tr(\Sigma_0^{-1}\Sigma_{AR}\Sigma_{MA}) + -p - \log\frac{|\Sigma_{AR}\Sigma_{MA}|}{|\Sigma_0|}\} \tag{64}$$

$$KLD = \frac{1}{2}\{p\theta^2 + 2(p-1)\theta\phi - (p-1)\log(1-\phi^2) - \log(1-\theta^2)\} \tag{65}$$

Here approximation $\log(1-x) \approx -x$ is used (for high values of p), which gives

$$KLD = \frac{1}{2}\{(p-1)(\phi+\theta)^2 - 2\theta^2\} \tag{66}$$

And for high values of p:

$$KLD = \frac{1}{2}\{(p-1)(\phi+\theta)^2\} \tag{67}$$

$$d(\phi,\theta) = \sqrt{(p-1)}|\theta+\phi| \tag{68}$$

Based on the formulated distance at equation 68, plots are generated for comparing the actual distance and formulated distance between the base model and the flexible model. R code at Appendix A.2 generates these plots of comparison. Actual distance and formulated distance comparison plots are shown the Figure 18 to the Figure 22:

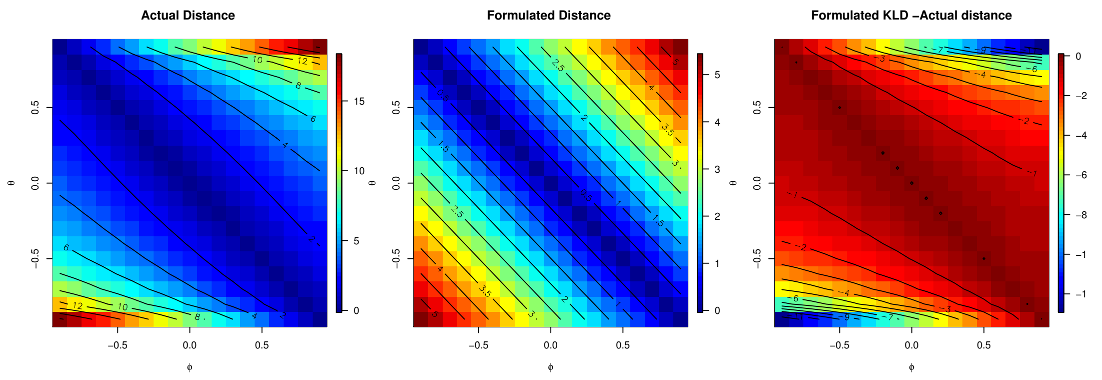

Figure 18: Comparison between the actual distance considering (Σ_{ARMA}) with the formulated distance (KLD based) when $p = 10$

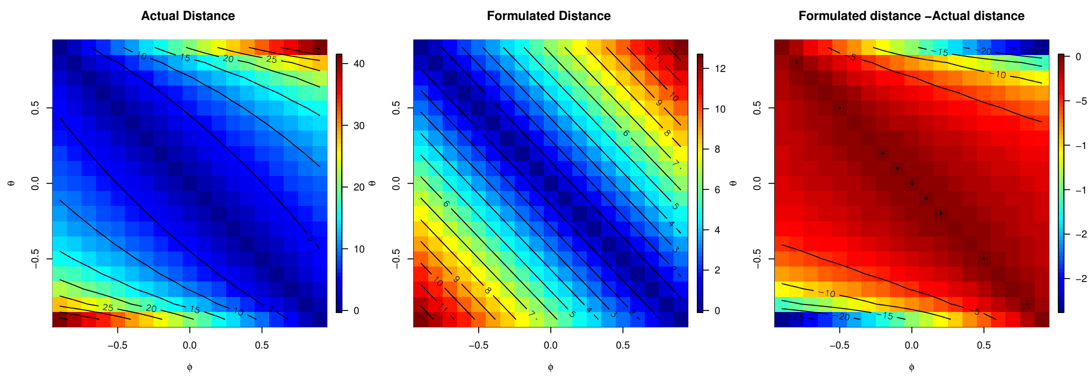

Figure 19: Comparison between the actual distance considering (Σ_{ARMA}) with the formulated distance (KLD based) when $p = 50$

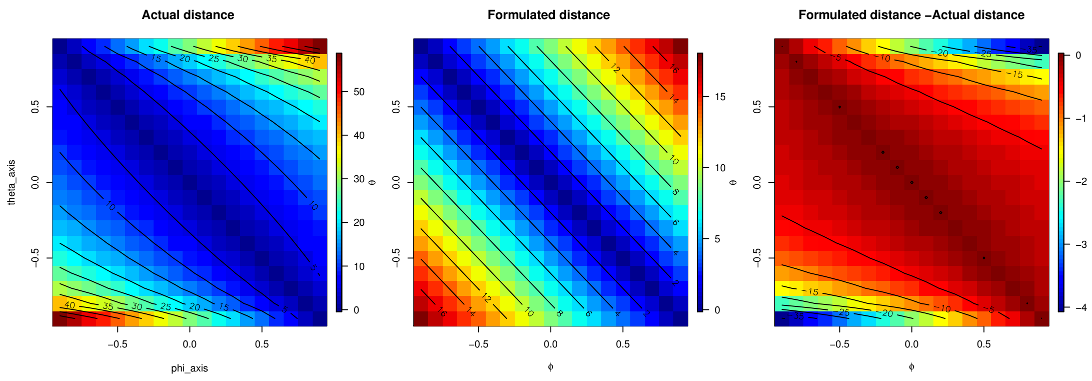

Figure 20: Comparison between the actual distance considering (Σ_{ARMA}) with the formulated distance (KLD based) when $p = 100$

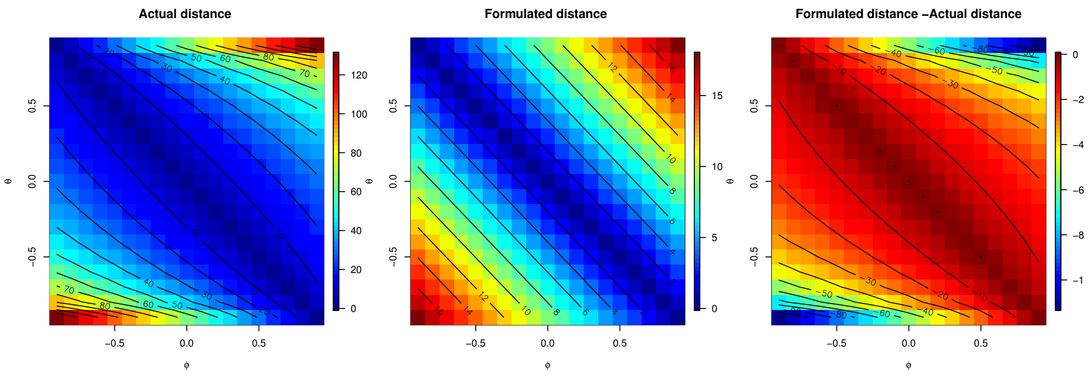

Figure 21: Comparison between the actual distance considering (Σ_{ARMA}) with the formulated distance (KLD based) when $p = 500$

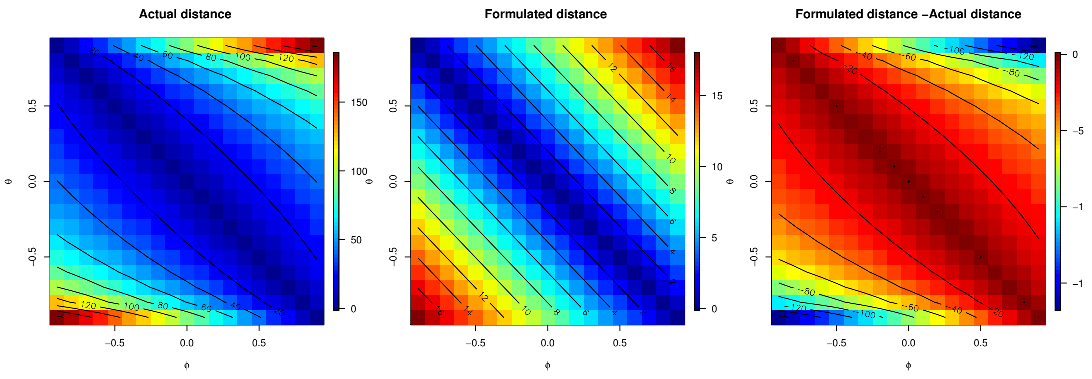

Figure 22: Comparison between the actual distance considering (Σ_{ARMA}) with the formulated distance (KLD based) when $p = 1000$

It is observed from the Figure 18 to the Figure 22 that for the lower values of p (number of data points), the formulated distance represent the actual distance well for almost all the points of the domain. However, for the higher values of p, the formulated distance does not fully represents the nature of the actual distance, to be more precise it is a good approximation around the base model. But for the flexible model far away from the base model the approximation does not work well, it is also observed that the KLD based distance between the base model and the flexible model increases rapidly if Σ_{ARMA} is used where as the formulated distance between base model and flexible model grows much slower.

The main reason for this behavior is that the formulation of the KLD based distance, uses more level of approximations on the initial approximation ($\Sigma_{ARMA} \approx \Sigma_{AR}\Sigma_{MA}$), and in the process of more approximations, the measure of the divergence between the base model and the flexible model looses it shape from elliptical to circular. And due to this reason, the actual distance grows much faster than the formulated distance. However, in the vicinity of the base model the formulated distance works well for higher p also.

The notion of the distance is found as a function of (ϕ, θ) by using approximation, construction of joint PC prior is done at equation 75:

$$\pi(d(\xi)) = \frac{\lambda \exp(-\lambda d(\xi))|detJ(\xi)|}{\mathcal{S}_{\lceil(\xi)}} \tag{69}$$

where
$J_{ij} = \frac{\partial d_i}{\partial d_j}$ and $\mathcal{S}_{\lceil(\xi)}$ length of the level sets of equidistant points. In this case

$$\xi = (\phi, \theta) \tag{70}$$

$$d(\xi) = d(\phi, \theta) = \sqrt{((p-1)}|\theta + \phi| \tag{71}$$

$$|detJ(\xi)| = (p-1) \tag{72}$$

$$\mathcal{S}_{\lceil(\xi)} = 2\sqrt{2}(2 - |\theta + \phi|) \tag{73}$$

$$\pi(\phi, \theta) = \frac{\lambda(p-1)\exp(-\lambda\sqrt{((p-1)}|\theta + \phi|)}{2\sqrt{2}(2 - |\theta + \phi|)} \tag{74}$$

$$\pi(\phi, \theta) = \frac{\lambda_0 \exp(-\lambda_1 |\theta + \phi|)}{(2 - |\theta + \phi|)} \tag{75}$$

where we have limitation $|\theta| < 1$, $|\phi| < 1$,

$$\begin{aligned} \lambda_1 &= \sqrt{((p-1)}\lambda \\ \lambda_0 &= \frac{\lambda(p-1)}{2\sqrt{2}C} = \frac{1}{12.78} \end{aligned} \tag{76}$$

where

$$C = \int_{-1}^{+1}\int_{-1}^{+1} \pi(\phi, \theta) d\phi d\theta = \frac{12.78\lambda(p-1)}{2\sqrt{2}} \tag{77}$$

Plots of the joint PC prior for ARMA (1,1) process for three different values of the scaling parameter $\lambda_1 = c(.1, 1, 10)$ are shown in the Figure 23 to Figure 25 :

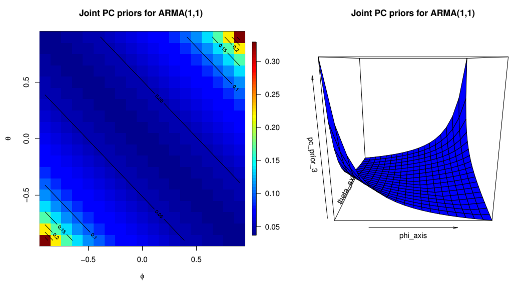

Figure 23: Joint PC priors of dependencies (ϕ, θ) of ARMA(1,1) process when the scaling parameter $\lambda_1 = .1$

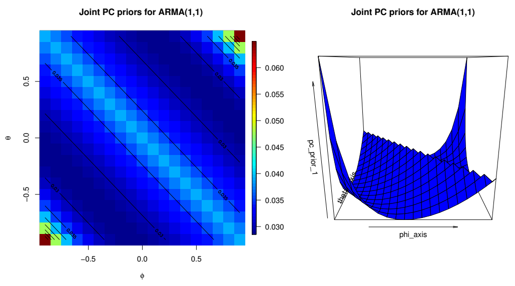

Figure 24: Joint PC priors of dependencies (ϕ, θ) of ARMA(1,1) process when the scaling parameter $\lambda_1 = 1$

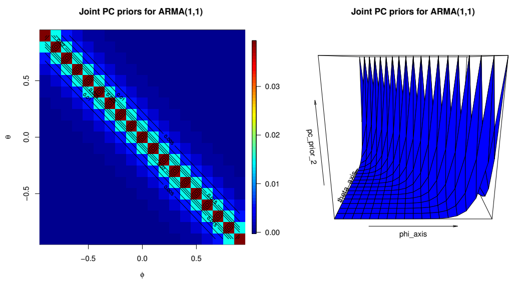

Figure 25: Joint PC priors of dependencies (ϕ, θ) of ARMA(1,1) process when the scaling parameter $\lambda_1 = 10$

For different values of the scaling parameter λ_1, the joint PC prior behaves both like an informative prior and a flat prior. From the Figure 23 to the Figure 25, it is observed that when the scaling parameter $\lambda_1 = 10$, the joint PC prior shows shrinkage and becomes informative whereas in the case when it was set to .1 the prior behaves like a flat prior.

As it is seen that the joint prior distribution leads to an inexact integration, hence marginalization analytically is not a fruitful idea.

4.3.2 Construction of the joint PC priors for the dependencies of the ARMA(1,1) process for the Base Model: AR(1) model

Let's say p is the dimension of the data-set, and the base model is considered to be an AR(1) process with parameter ϕ and the flexible model is considered an ARMA(1,1) process with parameter (ϕ, θ). For calculating the distance $d(\phi, \theta)$ based on the KLD, values of μ_0 and μ_{ARMA} and Σ_0 and Σ_{ARMA} are required. In this case $\mu_0 = \mu_{ARMA} = 0$ and $\Sigma_0 = \Sigma_{AR}$. Since, it is difficult to formulate the the distance (based on the KLD measure) between the base model and the flexible model analytically using Σ_{ARMA}, as it requires the value determinant of Σ_{ARMA} to be formulated in the closed form, however determinant of Σ_{ARMA} doesn't have a closed form, determinant of Σ_{ARMA} depends on the dimension of data, and numerical methods are required to find the determinant of Σ_{ARMA}. Hence, the approximation $\Sigma_{ARMA} \approx \Sigma_{AR}\Sigma_{MA}$ is suggested to use here as well. However, the approximation $\Sigma_{ARMA} \approx \Sigma_{AR}\Sigma_{MA}$ needs to be verified with reference to the measure of the distance between the base model and the flexible model.

For verification of the approximation, the distance comparison needs to be studied between the actual distance and the approximated distance between the base model and the flexible model. The approximated distance, between the base model and the flexible model is calculated using the approximation that $\Sigma_{ARMA} \approx \Sigma_{AR}\Sigma_{MA}$.

Distance comparisons plots between the actual and the approximated distances are shown in the Figure 26 to the Figure 30:

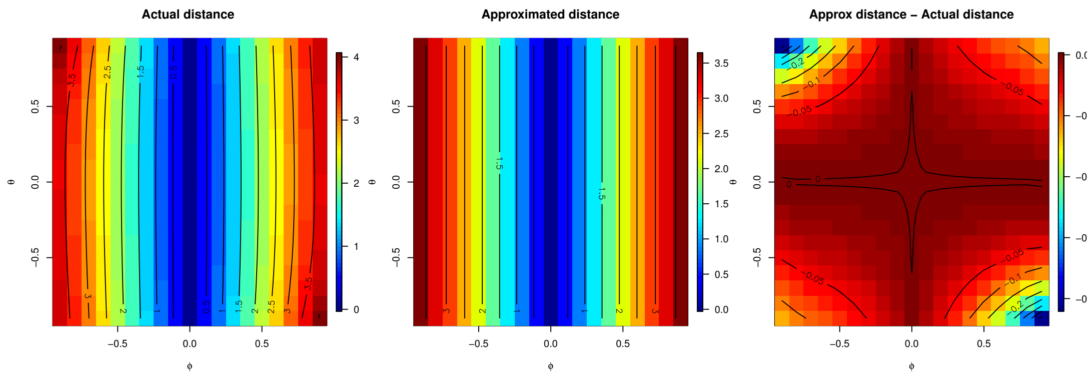

Figure 26: Comparison between the actual distance (considering Σ_{ARMA}) with the approximated distance (considering $\Sigma_{AR}\Sigma_{MA}$) when $n = 10$

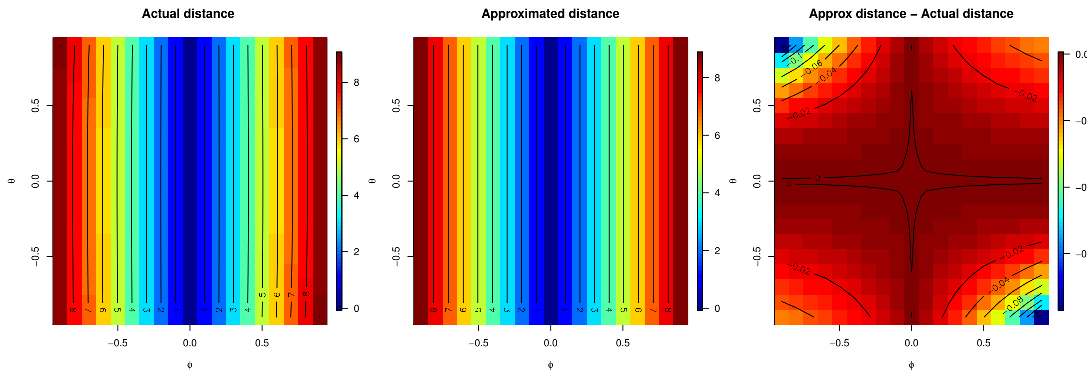

Figure 27: Comparison between the actual distance (considering Σ_{ARMA}) with the approximated distance (considering $\Sigma_{AR}\Sigma_{MA}$) when $n = 50$

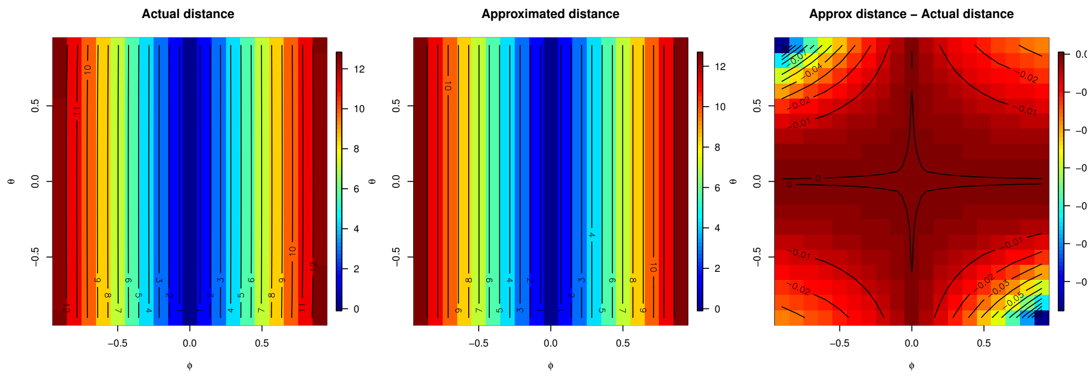

Figure 28: Comparison between the actual distance (considering Σ_{ARMA}) with the approximated distance (considering $\Sigma_{AR}\Sigma_{MA}$) when $n = 100$

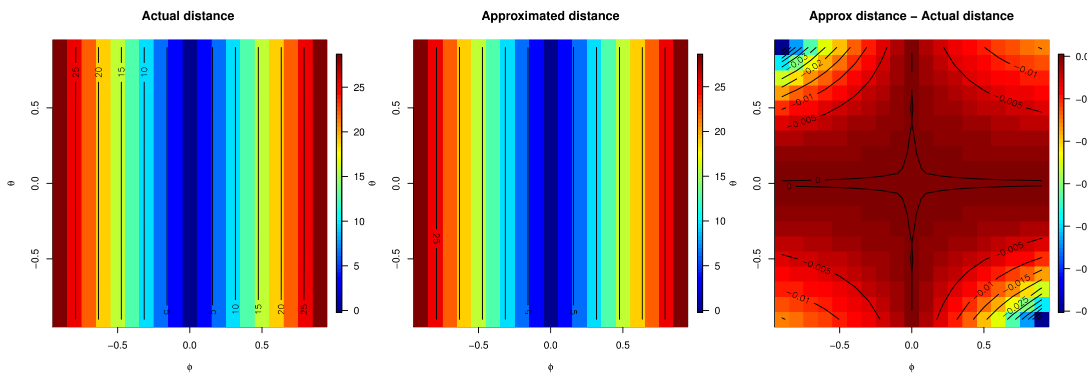

Figure 29: Comparison between the actual distance (considering Σ_{ARMA}) with the approximated distance (considering $\Sigma_{AR}\Sigma_{MA}$) when $n = 500$

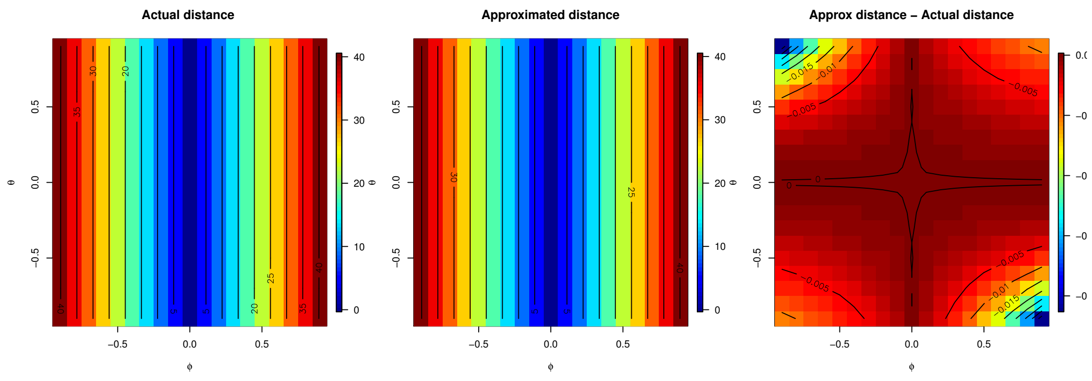

Figure 30: Comparison between the actual distance (considering Σ_{ARMA}) with the approximated distance (considering $\Sigma_{AR}\Sigma_{MA}$) when $n = 1000$

It is observed from the Figure 26 to the Figure 30, that the approximation works pretty well for almost every point of the domain, except very high values of the parameters. With higher number of points in data set, the quality of the approximation improves significantly. This is observed from looking at the contour plots of the difference between the approximated distance and the actual distance. Hence, the approximation proposed is used for constructing the PC prior in this case:

$$KLD(\mathcal{N}_1^p||\mathcal{N}_0^p) = \frac{1}{2}\{tr(\Sigma_0^{-1}\Sigma_1) + (\mu_0 - \mu_1)^T\Sigma_0^{-1}(\mu_0 - \mu_1) - p - \log\frac{|\Sigma_1|}{|\Sigma_0|}\} \tag{78}$$

$$KLD(\theta|\phi) = \frac{1}{2}\{tr(\Sigma_{AR}^{-1}\Sigma_{ARMA}) - p - \log\frac{|\Sigma_{ARMA}|}{|\Sigma_{AR}|}\} \tag{79}$$

$$KLD(\theta|\phi) = \frac{1}{2}\{tr(\Sigma_{AR}^{-1}\Sigma_{AR}\Sigma_{MA}) - p - \log\frac{|\Sigma_{AR}\Sigma_{MA}|}{|\Sigma_{AR}|}\} \tag{80}$$

$$KLD(\theta|\phi) = KLD(MA(1)_\theta) \tag{81}$$

which will lead to the PC prior formulated at Section4.2. It can be summarized by equation 82:

$$\pi(\theta|\phi) = \pi_{PC}(\theta) \tag{82}$$

It is observed that if the base model is assumed to be an AR(1) process and the ARMA(1,1) process represents the flexible model, then the joint PC prior for the dependencies of the ARMA(1,1) process is given by equation 83:

$$\pi_{PC}(\phi,\theta) = (\pi_{PC}(\phi), \pi_{PC}(\theta)) \tag{83}$$

4.3.3 Construction of the joint PC priors for the dependencies of the ARMA(1,1) process for the Base Model: MA(1) model

Let's say p is the dimension of the data-set, and the base model is considered to be an MA(1) process with the parameter θ and the flexible model is considered an ARMA(1,1) process with the parameters (ϕ, θ). For calculating the distance $d(\phi, \theta)$ based on the KLD, values of μ_0 and μ_{ARMA} and Σ_0 and Σ_{ARMA} are required. In this case $\mu_0 = \mu_{ARMA} = 0$ and $\Sigma_0 = \Sigma_{MA}$. Since, it is difficult to formulate the the distance (based on the KLD measure) between the base model and the flexible model analytically using Σ_{ARMA}, as it requires the value determinant of Σ_{ARMA} to be formulated in the closed form, however determinant of Σ_{ARMA} doesn't have a closed form, determinant of Σ_{ARMA} depends on the dimension of the data-set, and numerical methods are required to find the determinant of Σ_{ARMA}. Hence, the approximation $\Sigma_{ARMA} \approx \Sigma_{AR}\Sigma_{MA}$ is suggested to use here as well. However, the approximation $\Sigma_{ARMA} \approx \Sigma_{AR}\Sigma_{MA}$ needs to be verified with reference to the measure of distance between the base model and the flexible model.

For verification of the approximation, the distance comparison needs to be studied between the actual distance and the approximated distance between the base model and the flexible model. The approximated distance, between the base model and the flexible model is calculated using the approximation that $\Sigma_{ARMA} \approx \Sigma_{AR}\Sigma_{MA}$.

Distance comparisons plots between the actual and the approximated distances are shown in Figure 31 to Figure 35:

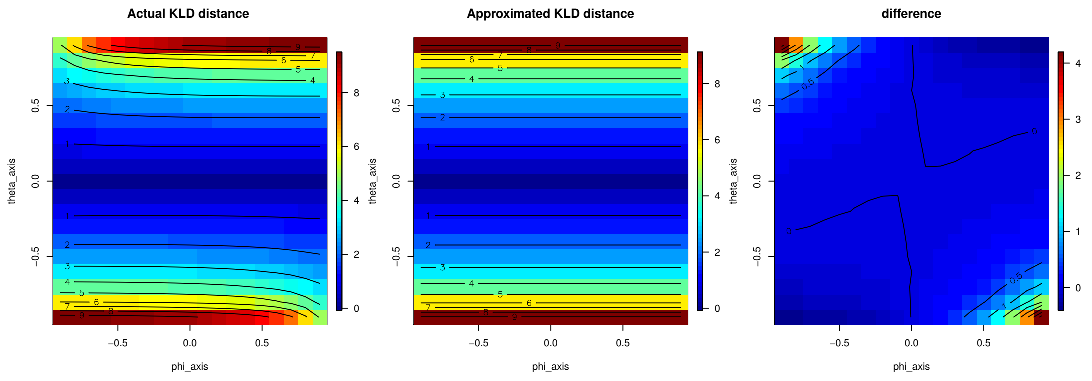

Figure 31: Comparison between the actual distance (considering Σ_{ARMA}) with the approximated distance (considering $\Sigma_{AR}\Sigma_{MA}$) when $n = 10$

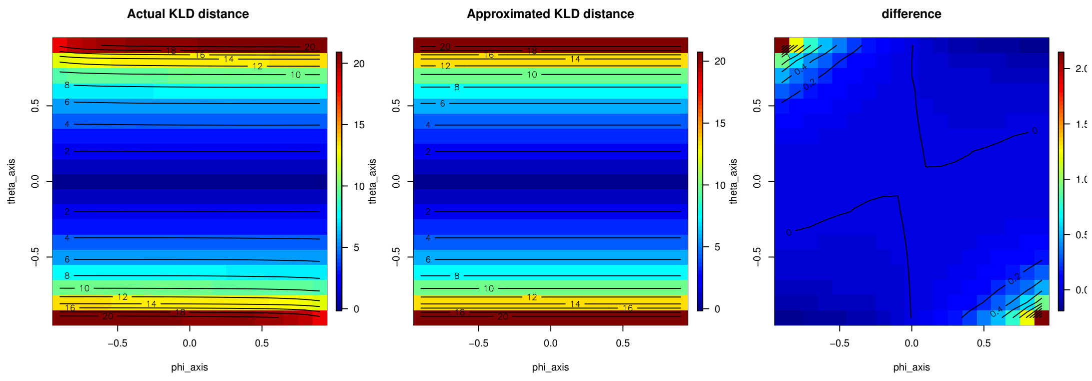

Figure 32: Comparison between the actual distance (considering Σ_{ARMA}) with the approximated distance (considering $\Sigma_{AR}\Sigma_{MA}$) when $n = 50$

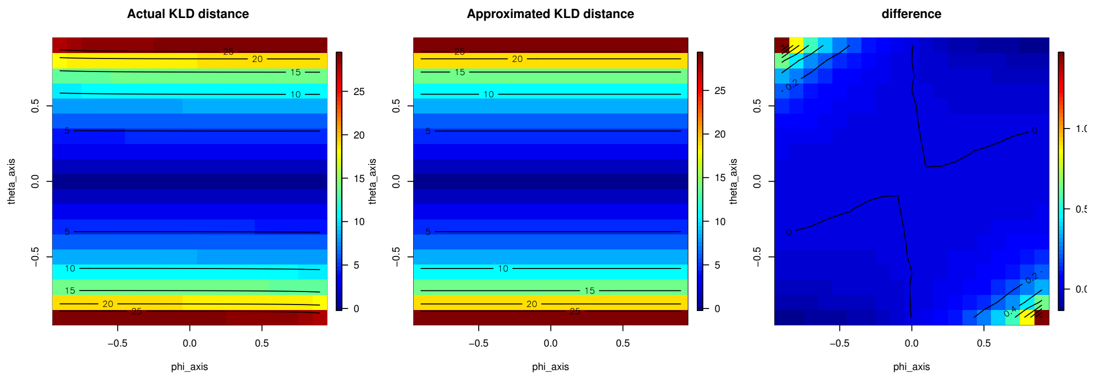

Figure 33: Comparison between the actual distance (considering Σ_{ARMA}) with the approximated distance (considering $\Sigma_{AR}\Sigma_{MA}$) when $n = 100$

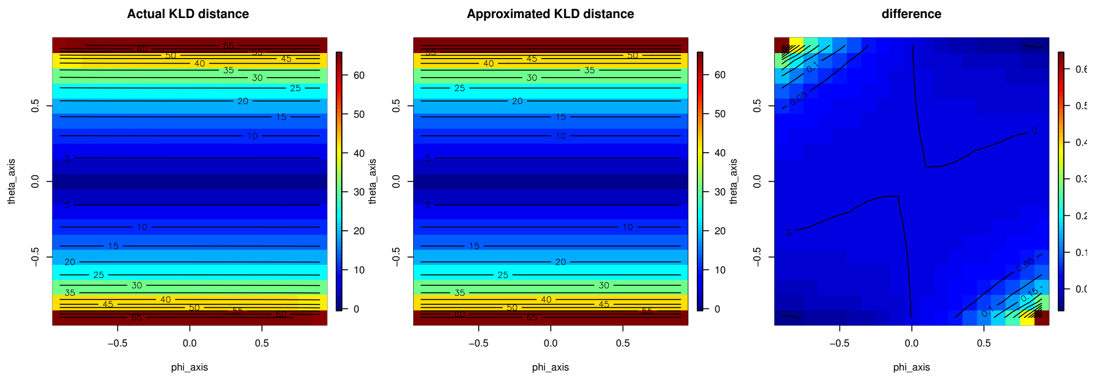

Figure 34: Comparison between the actual distance (considering Σ_{ARMA}) with the approximated distance (considering $\Sigma_{AR}\Sigma_{MA}$) when $n = 500$

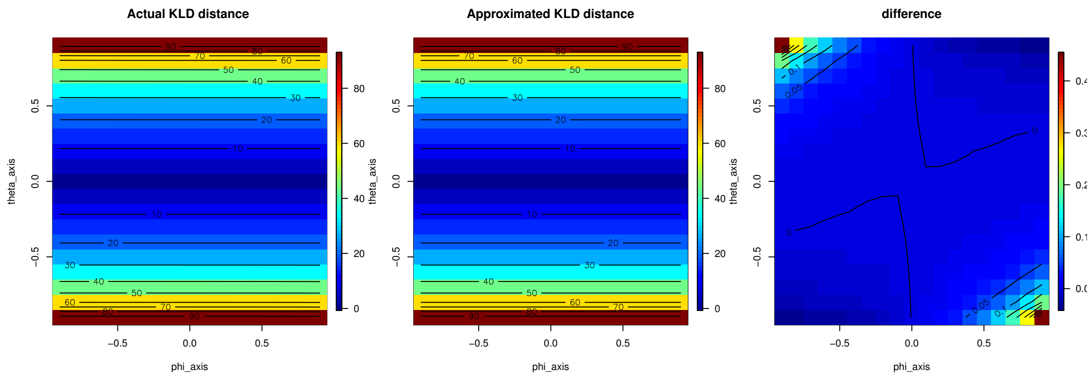

Figure 35: Comparison between the actual distance (considering Σ_{ARMA}) with the approximated distance (considering $\Sigma_{AR}\Sigma_{MA}$) when $n = 1000$

It is observed from Figure 31 to Figure 35, that the approximation works pretty well for almost every point of the domain, except very high values of the parameters. With higher number of points in the data set, the quality of the approximation improves significantly.This is observed by looking at the contour plots of the difference between the approximated distance and the actual distance. Hence, the approximation proposed is used for constructing the PC prior in this case:

$$KLD(\mathcal{N}_1^p||\mathcal{N}_0^p) = \frac{1}{2}\{tr(\Sigma_0^{-1}\Sigma_1) + (\mu_0 - \mu_1)^T\Sigma_0^{-1}(\mu_0 - \mu_1) - p - \log\frac{|\Sigma_1|}{|\Sigma_0|}\} \tag{84}$$

$$KLD(\phi|\theta) = \frac{1}{2}\{tr(\Sigma_{MA}^{-1}\Sigma_{ARMA}) - p - \log\frac{|\Sigma_{ARMA}|}{|\Sigma_{MA}|}\} \tag{85}$$

$$KLD(\theta|\phi) = \frac{1}{2}\{tr(\Sigma_{MA}^{-1}\Sigma_{MA}\Sigma_{AR}) - p - \log\frac{|\Sigma_{AR}\Sigma_{MA}|}{|\Sigma_{MA}|}\} \tag{86}$$

$$KLD(\theta|\phi) = KLD(AR(1)_\phi) \tag{87}$$

which will lead to the PC prior formulated at Section4.1.1. It can be summarized by equation 88:

$$\pi(\phi|\theta) = \pi_{PC}(\phi) \tag{88}$$

It is observed that if the base model is assumed to be an MA(1) process and the ARMA(1,1) process represents the flexible model, then the joint PC prior for the dependencies of the ARMA(1,1) process is given by equation 89:

$$\pi_{PC}(\phi,\theta) = (\pi_{PC}(\phi), \pi_{PC}(\theta)) \tag{89}$$

5 Application of the PC priors in the INLA framework

The PC priors have been constructed analytically in the Section 4, for various time series models, in this section, the PC priors developed are used as prior distribution in the INLA (Rue at el 2009)[18] framework. In this section some time series data sets have been simulated and fitted using the PC priors.

5.1 Fitting of the data-set in case of an AR(1) process

The PC prior for an AR(1) process are developed in Section4.1.1, in this section, simulated data-set of an AR(1) process are fit using the PC prior distributions in the INLA framework.

Effect of the the user defined scale when the PC priors are constructed assuming the base model to be iid $(\Phi = 0)$**:**

The time series data-set of an AR(1) process is simulated using the $\phi = 0.4$ mixed with the error signal. The standard deviation (σ_e) of the error signal is set to .1. Different user defined scales are chosen to fit the data and to observe the effect of user defined scales on the PC priors, we have considered three user defined scales, probability statement in this case is given by equation 90

$$\Pr(|\phi| > U) = \alpha \tag{90}$$

We have defined three scales for $\alpha = (0.5, 0.05, 0.005)$, where the upper bound U is set to 0.4. R Code at Appendix A.3 is executed in this case and following estimates are obtained:

```
>r1$summary.hyperpar
                mean        sd 0.025quant  0.5quant 0.975quant      mode
Rho for z 0.4714221 0.1070131  0.2316688 0.4826291  0.6483609 0.5058364



> r2$summary.hyperpar
                mean        sd 0.025quant 0.5quant 0.975quant      mode
Rho for z 0.3842134 0.1259087  0.1105257 0.395138  0.5988609 0.4208695

> r3$summary.hyperpar
                mean        sd 0.025quant  0.5quant 0.975quant      mode
Rho for z 0.2769954 0.1357531 0.02481314 0.2811809  0.5297283 0.3031576
```

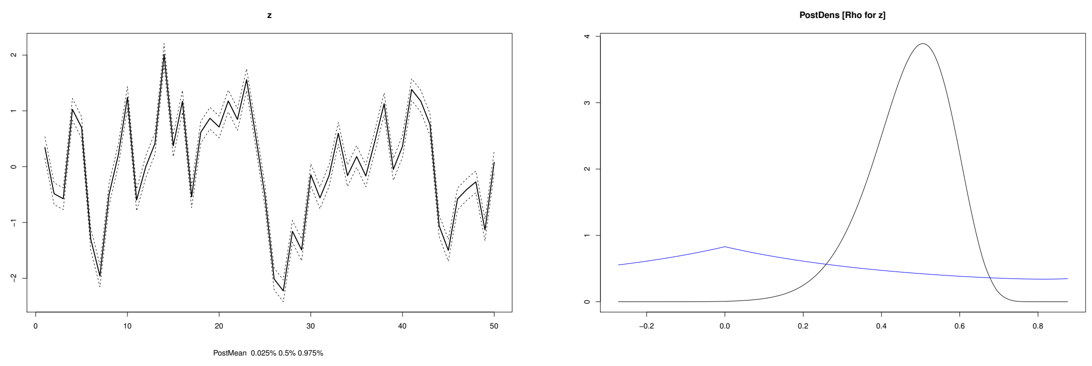

Figure 36: Plot of PC Priors(in Blue) when user defined scales are $U = 0.4$ and $\alpha = 0.5$

It is seen from the Figure 36 that the PC Prior behaves like a flat prior, as weakly informative user defined scale is used. It has allowed 50% of probability mass for ϕ away from the maximum limit of 0.4 for ϕ.

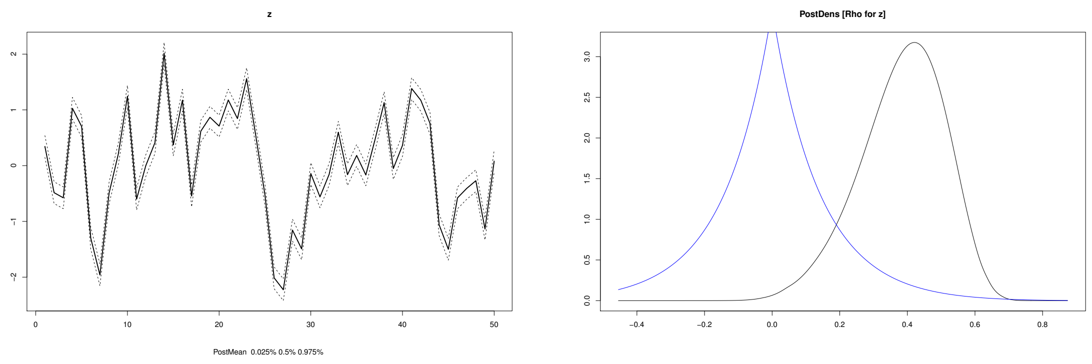

Figure 37: Plot of the PC Priors(in Blue) when the user defined scales are $U = 0.4$ and $\alpha = 0.05$

When the user defined scale is changed to the more informative condition i.e. set $\alpha = 0.05$, resulting the plots in the Figure 37. It is seen that the PC Prior shows excellent shrinkage towards the base model. It is also observed that the estimates for ϕ are closer towards the base model as the prior has shown shrinkage towards the base model.

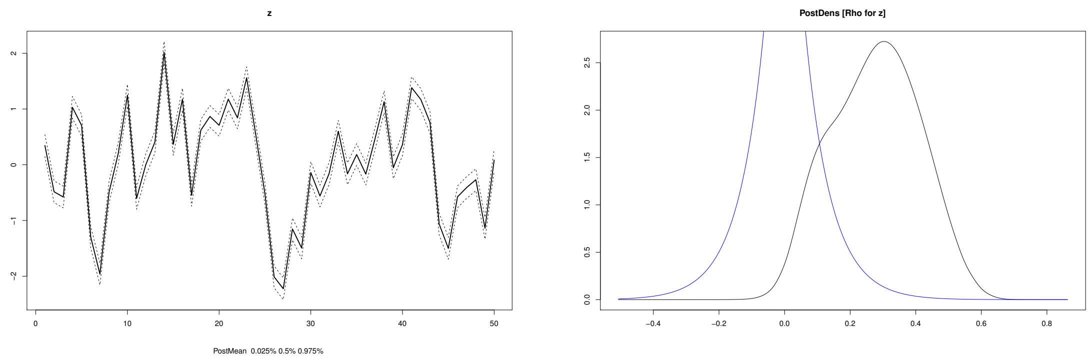

Figure 38: Plot of the PC Priors(in Blue) when the user defined scales are $U = 0.4$ and $\alpha = 0.005$

In this case only 0.5 % probability mass of ϕ is allowed to be away from the maximum limit of 0.4. Stronger shrinkage is expected for the prior distribution and the estimates for ϕ will be more close to the base models, as the PC prior in this case is highly informative. Plots are shown in the Figure 38 in this case. It is observed that the PC Priors are exhibiting the strong shrinkage towards the base models and the estimates for ϕ are closer towards the base model.

We observed that the PC Priors can be both highly informative or non informative based on the idea we have about the data set/process. The estimation for the parameter of interest depends on the prior distribution. It should be noted that the estimation will be more towards the base model if the prior distribution has the stronger shrinkage towards the base model.

Effect of the user defined scale when the PC priors are constructed assuming the base model to be No change in time ($\phi = 1$):

The time series data-set of an AR(1) process, is simulated using the $\phi = 0.95$ mixed with the error signal. The standard deviation (σ_e) of the error signal is set to .1. Different user defined scales are chosen to fit the data and observe the effect of user defined scales on the PC priors, we have considered three user defined scales, probability statement in this case is given by equation 91

$$\Pr(|\phi| > U) = \alpha \tag{91}$$

We have defined three scales for $\alpha = (0.3, 0.7, 0.9)$, where is the upper bound U is set to 0.95. R Code at Appendix A.4 is executed in this case and following estimates are obtained:

```
> r11$summary.hyperpar
               mean         sd 0.025quant  0.5quant 0.975quant      mode
Rho for z 0.9296304 0.01739504  0.8891834 0.9319316  0.9568544 0.9361719
> r12$summary.hyperpar
               mean         sd 0.025quant  0.5quant 0.975quant      mode
Rho for z 0.9318041 0.01659387   0.893326 0.9339559  0.9579153 0.9379378
> r13$summary.hyperpar
               mean         sd 0.025quant  0.5quant 0.975quant      mode
Rho for z 0.9342175 0.01572893  0.8978609 0.9362119  0.9591127 0.9399193
```

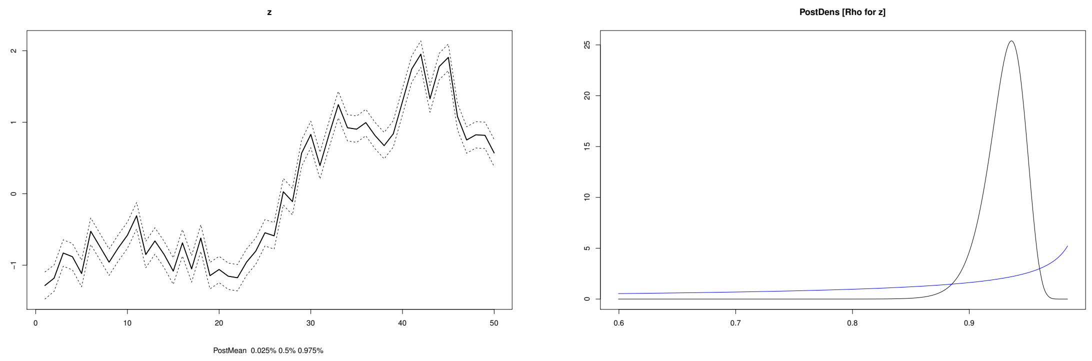

Figure 39: Plot of the PC Priors(in Blue) when the user defined scales are $U = 0.95$ and $\alpha = 0.3$

It is seen from Figure 39 that probability mass of the prior distribution for ϕ is concentrated between .9 and .95, since the mild conditions are put on the prior, in this case that only 70% probability mass can stay below the $U = .95$

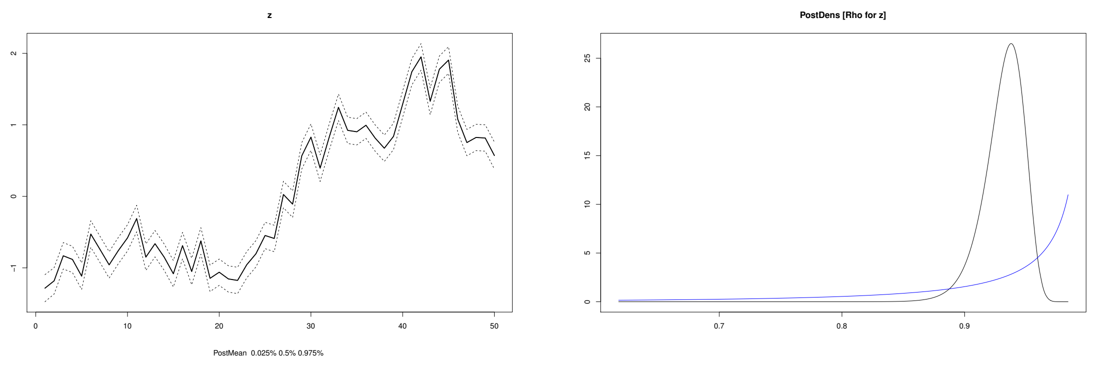

Figure 40: Plot of the PC Priors(in Blue) when the user defined scales are $U = 0.95$ and $\alpha = 0.7$

In the Figure 40, more strong user defined scale is considered for constructing the PC Prior, only 30% probability mass of ϕ can stay below the $U = .95$. So the estimation for ϕ is closer to the base model.

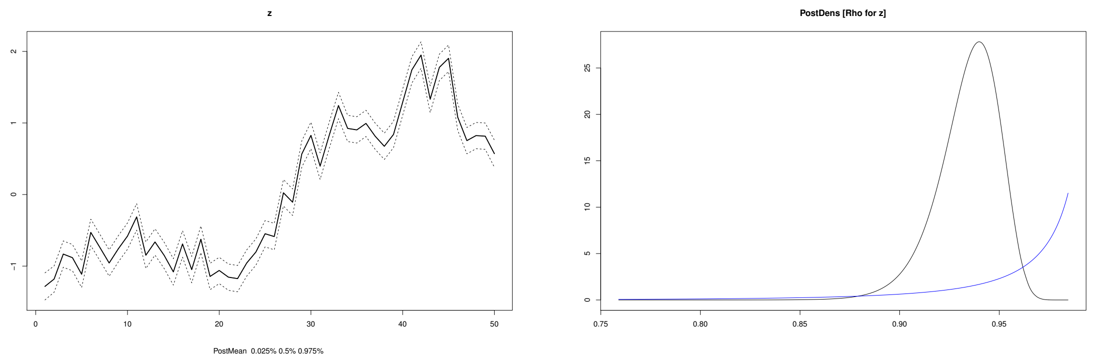

Figure 41: Plot of PC the Priors(in Blue) when the user defined scales are $U = 0.95$ and $\alpha = 0.9$

In the Figure 41, stronger user defined scale is considered for constructing the PC prior, only 10% probability mass of ϕ can stay below the 0.95. So the estimation for ϕ is closer to the base model.

It is observed that the PC priors are sensitive towards the user defined scales. So, the better user defined scale (the better idea we have about the process), will lead priors to become more informative. More informative priors show excellent shrinkage towards the base model. Based on the degree of informativeness of the prior distributions, the estimation of the parameter will be more towards the base model. It is worth mentioning that the PC priors may behave like flat priors or uninformative priors if the suitable user defined scales are used for construction, in those cases estimation of the parameter will be governed by the likelihood.

5.2 Fitting of the data-set in case of an MA(1) process

The PC prior for dependency at lag one for an MA(1) process is constructed analytically in Section 4.2. In this section, simulated data-set of an MA(1) process are fit using the PC prior distributions in the INLA framework.

Effect of the user defined scale when the PC prior are constructed assuming the base model to be iid i.e. $(\theta = 0)$**:**

The time series data-set of an MA(1) process is simulated using the $\theta = 0.4$ mixed with the error signal. The standard deviation (σ_e) of the error signal is set to 1. Different user defined scales are chosen to fit the data and observe the effect of user defined scales on the PC priors, we have considered three user defined scales, probability statement in this case is given by equation 92

$$\Pr(|\theta| > U) = \alpha \tag{92}$$

R Code at Appendix A.5 is executed in this case.

Case-I: when user defined scales are $U = 0.4$ and $\alpha = 0.5$

It is observed from the Figure 42 that the PC Priors behaves like a flat prior, since weakly informative user defined scales are used to construct them. User defined scales allowed only 50% of probability mass for θ in the limit formed by $-0.4, 0.4$.

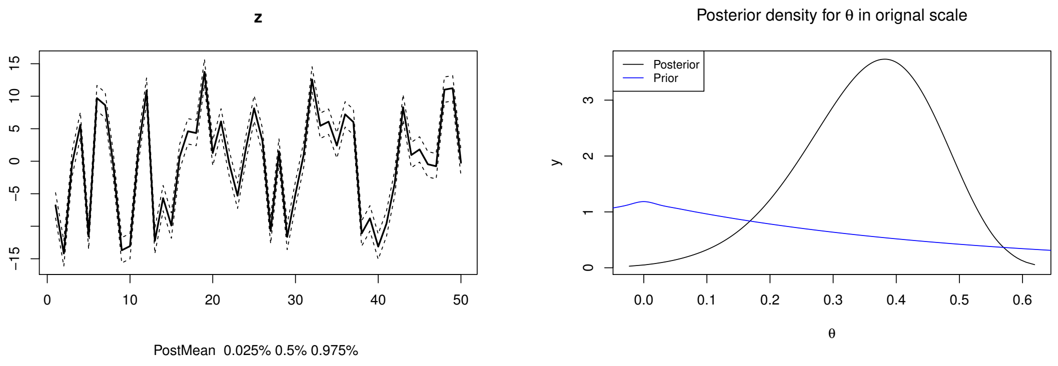

Figure 42: Posteriors using the PC priors when U=.4 and alpha=.5

In this case, estimates for θ are:

```
> summary(r1);
Call:
c("inla(formula = formula, family = \"gaussian\",
data = data.frame(y, ",  "    z = 1:n),
verbose = TRUE,
control.family = list(hyper = list
(prec = list(initial = log(1/s^2), ", " fixed = TRUE))))")

Time used:
 Pre-processing    Running inla Post-processing           Total
         5.2401         12.0182          0.4688         17.7271

The model has no fixed effects
Random effects:

Name   Model
 z   RGeneric2

Model hyperparameters:
               mean    sd 0.025quant 0.5quant 0.975quant   mode
```

```
Theta1 for z 0.7508 0.241     0.2538   0.7599      1.195 0.7797

Expected number of effective parameters(std dev): 49.34(0.1001)

Number of equivalent replicates:1.013

Marginal log-Likelihood:  -188.21
> r11=inla.hyperpar(r1);
> 2/(1+exp(-r11$summary.hyperpar$mean))-1
[1] 0.3592295
```

Case-II: when user defined scales are $U = 0.4$ and $\alpha = 0.05$
In this case, the user defined scale are changed to the a more informative condition i.e. set to $\alpha = 0.05$, resulted the plots in the Figure 43. It is observed that the PC Prior shows excellent shrinkage towards the base model. Since, the prior is more informative, the estimates for θ are closer towards the base model.

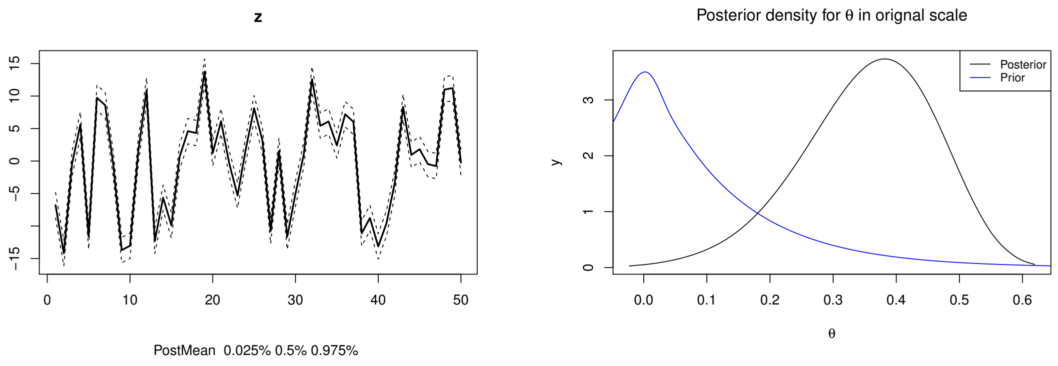

Figure 43: Posteriors using the PC priors when U=.4 and alpha=.05

In this case, estimates for θ are:

```
> summary(r2)

Call:
c("inla(formula = formula, family = \"gaussian\",
data = data.frame(y, ",  " z = 1:n),
control.family = list(hyper = list
(prec = list(initial = log(1/s^2), ",  " fixed = TRUE))))")

Time used:
 Pre-processing    Running inla Post-processing           Total
         2.8706          5.8753          0.2049          8.9508

The model has no fixed effects
Random effects:
Name   Model
 z   RGeneric2

Model hyperparameters:
              mean     sd 0.025quant 0.5quant 0.975quant   mode
```

```
Theta1 for z 0.5973 0.2512     0.1092   0.6003      1.076 0.6137

Expected number of effective parameters(std dev): 49.40(0.0848)

Number of equivalent replicates : 1.012

Marginal log-Likelihood:  -188.77
> r12=inla.hyperpar(r2)
> 2/(1+exp(-r12$summary.hyperpar$mean))-1
[1] 0.2886385
```

Case-III: when user defined scales are $U = 0.4$ and $\alpha = 0.005$

The user defined scales are changed to a more informative condition i.e. are set to $\alpha = 0.005$ in this case, resulted the plots in the Figure 44. It is observed that the PC prior shows excellent shrinkage towards the base model. Since the prior is more informative in this case, the estimates for θ are closer towards the base model.

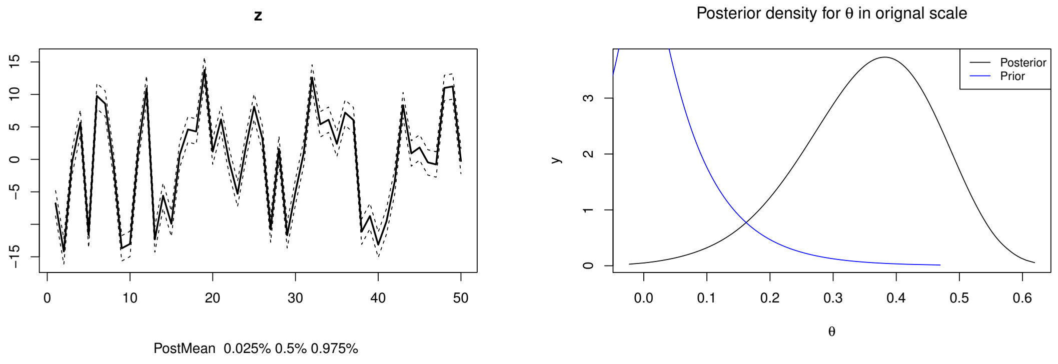

Figure 44: Posteriors using the PC priors when U=.4 and alpha=.005

In this case, estimates for θ are:

```
> summary(r3)
Call:
c("inla(formula = formula, family = \"gaussian\",
data = data.frame(y, ",  " z = 1:n),
control.family = list(hyper = list(
prec = list(initial = log(1/s^2), ",  " fixed = TRUE))))")

Time used:
 Pre-processing    Running inla Post-processing           Total
         2.6066          5.4645          0.2339          8.3050
The model has no fixed effects
Random effects:
Name   Model
z   RGeneric2

Model hyperparameters:
               mean     sd 0.025quant 0.5quant 0.975quant mode
Theta1 for z 0.4277 0.2351     0.0204   0.4113     0.9093 0.32
```

```
Expected number of effective parameters(std dev): 49.44(0.0623)

Number of equivalent replicates : 1.011

Marginal log-Likelihood:  -189.54
> r13=inla.hyperpar(r3)
> 2/(1+exp(-r13$summary.hyperpar$mean))-1
[1] 0.209396
```

It is observed that the PC priors are sensitive towards the user defined scales. The better user defined scales (the better idea we have about the process), will lead priors to become more informative. More informative priors show excellent shrinkage towards the base model. Based on the degree of informativeness of the prior distribution, the estimation of the parameter will be more towards the base model. It is worth mentioning that the PC prior may behave like a flat prior or an uninformative prior if the suitable user defined scales are used for constructing them, in those cases the estimation of the parameter will be governed by likelihood.

6 Conclusions and Recommendations

Conclusions

Prior selection is a crucial issue in the Bayesian setting. Priors are the strength of Bayesian statistical inference at the same time the method of prior selections becomes a point of critical discussions. In this report, we have explained a systematic approach to construct the prior for the parameter of interest for some time series models. Key for constructing the PC prior lies in the way the base model is selected. As it is seen in the case of an AR(1) process that different base models result into different shapes of the prior distribution, the same has also become evident when the different base models in case of the ARMA(1,1) process led to the different joint PC priors for the dependencies of the model. So the most important part of whole construction is to find a suitable base model for the given problem. There can be several approaches to select the base model and based on the selection of the base model the prior distribution for the hyper parameter will change. Another important factor while constructing the PC prior is that the distance based on the KLD should be formulated in such a way that the distance factorizes with respect to hyper parameters and number of points. In this way, the user defined scales take care of the scaling parameter for the particular PC prior. If the distance becomes implicit function with reference to number of points and hyper parameter, then constructing PC prior analytically will be quite difficult.

Recommendations

R -generic function for the PC priors developed for dependencies of the MA(1) process and of the ARMA(1,1) process can be implemented in the INLA framework. Based on the results in this report, the PC prior for the higher order MA process may be formulated. Better approximation method should be found for the ARMA(1,1) process so that support for prior distribution can be extended. PC prior for the non stationary AR processes and non invertible MA process should also be developed.

References

[1] Plummer, M. (2003). JAGS: A program for analysis of Bayesian graphical models using Gibbs sampling In Proceedings of the 3rd International Workshop on Distributed Statistical Computing (DSC 2003). March, pages 20–22

[2] Bernardo, J. M. (1979). Reference posterior distributions for Bayesian inference. Journal of the Royal Statistical Society, Series B, 41(2):113–147.

[3] Berger, J. O. (2006). The case for objective Bayesian analysis (with discussion). Bayesian Analysis, 1(3):385–402.

[4] Berger, J. O., Bernardo, J. M., and Sun, D. (2009). The formal definition of reference priors. The Annals of Statistics, 37(2):905–938.

[5] Ghosh, M. (2011). Objective priors: An introduction for frequentists (with discussion). Statistical Science, 26(2):187– 202.

[6] Goldstein, M. (2006). Subjective Bayesian analysis: Principles and practice (with discussion). Bayesian Analysis, 1(3):403–420

[7] Kamary, K. (2014). A discussion on Bayesian analysis: Selecting Noninformative Priors. arXiv preprint arXiv:1402.6257

[8] Gelman, A. (2006). Prior distributions for variance parameters in hierarchical models (with discussion). Bayesian analysis, 1(3):515–534.

[9] Gelman, A., Jakulin, A., Pittau, M. G., and Su, Y. S. (2008). A weakly informative default prior distribution for logistic and other regression models. The Annals of Applied Statistics, 2(4):1360–1383.

[10] Evans, M. and Jang, G. H. (2011). Weak informativity and the information in one prior relative to another Statistical Science, 26(3):423–439

[11] Polson, N. G. and Scott, J. G. (2012). On the half-Cauchy prior for a global scale parameter. Bayesian Analysis, 7(4):887–902.

[12] Simpsons D P Thiago G. M Riebler A Fugstad G A Rue H Sigrunn H. S. Penalising model component complexity: A principled, practical approach to construct priors (December 1, 2014).

[13] Sigrunn H. S Rue H Penalized complexity priors for stationary autoregressive processes (August 29, 2016)

[14] Brockwell, P. J. and Davis, R. A. (2002). Introduction to Time Series and Forecasting. SpringerVerlag, New Work, 2nd edition.

[15] Chatfield, C. (2003). The Analysis of Time Series: An Introduction. Chapman & Hall/CRC, 6th edition

[16] Prado R. and West, M. (2010). Time Series - Modeling, Computation and Inference. Chapman & Hall/CRC, Boca Raton

[17] Chinipardazthe R. and Cox T. F. Inverse of Covariance Matrices for the ARMA (p, q) class of process in example at 304

[18] Rue, H., Martino, S., and Chopin, N. (2009). Approximate Bayesian inference for latent Gaussian models by using integrated nested Laplace approximations (with discussion). Journal of the Royal Statistical Society, Series B, 71:319–392.

[19] Choi Almost All About Unit Roots: Foundations, Developments, and Applications Book page 108

[20] James O. B Ruo-young Y Noninformative Priors and Bayesian Testing for AR(1) Model Technical Report 92-45C, page 7

Appendices

A Appendix A.1

Following code contains R code for comparing actual distance and approximated distance for MA(1) process, generates curves for PC priors for MA(1) process.

```
require(MASS)
#require(sos)
library(INLA)
library(spam)
#library(MASS)

distance_comparision = function(n,theta) {


  sigma1 = matrix(0, nrow = n, ncol = n)
  sigma1_approx = matrix(0, nrow = n, ncol = n)
  for(i in 1:n) {

    if(i ==1)  {

      sigma1 [i,i] = 1
      sigma1 [i,i+1] = theta
      sigma1_approx [i,i] = 1+(theta)^2
      sigma1_approx [i,i+1] = theta

    }
    else if (i == n){
      sigma1 [i,i] = 1
      sigma1 [i,i-1] = theta
      sigma1_approx [i,i] = 1+(theta)^2
      sigma1_approx [i,i-1] = theta
    }
    else {
      sigma1 [i,i] = 1+(theta)^2
      sigma1 [i,i-1] = theta
      sigma1 [i,i+1] = theta

      sigma1_approx [i,i] = 1+(theta)^2
      sigma1_approx [i,i-1] = theta
      sigma1_approx [i,i+1] = theta
    }
  }

  Distance_actual = (sum(diag(sigma1)) - n - log(det(sigma1)))^.5
  Distance_approximated = (sum(diag(sigma1_approx)) - n -
                                  log(det(sigma1_approx)))^.5

  return(c(Distance_actual,Distance_approximated))

}

distances = matrix(0, nrow = 20, ncol = 2)

par(mfrow=c(2,3))


##### for theta = 0.1
distances = matrix(0, nrow = 20, ncol = 2)

for(n in seq(5,100 ,by = 5)){


  a= distance_comparision(n,0.1)
  distances[n/5,1] = a[1]
  distances[n/5,2]= a[2]

}
plot(seq(5,100 ,by = 5),distances[,1],xlab = "dimension of square matrix
     ",
     ylab = "KLD Distance",type="p",col="black",
     main = "Comaprision of distances for theta 0.1",cex.main = .6)

lines(seq(5,100 ,by = 5),distances[,2],col="green")

legend("topleft",
       legend=c("Actual distance", "Approximated distance"),
       col=c("black", "Green"),
       lty=c(1,1), cex=0.5)


##### for theta = 0.3
distances = matrix(0, nrow = 20, ncol = 2)

for(n in seq(5,100 ,by = 5)){


  a= distance_comparision(n,0.3)
  distances[n/5,1] = a[1]
  distances[n/5,2]= a[2]

}
plot(seq(5,100 ,by = 5),distances[,1],xlab = "dimension of square matrix
     ",
     ylab = "KLD Distance",type="p",col="black",
     main = "Comaprision of distances for theta 0.3",cex.main = .6)

lines(seq(5,100 ,by = 5),distances[,2],col="green")

legend("topleft", legend=c("Actual distance", "Approximated distance"),
       col=c("black", "Green"), lty=c(1,1), cex=0.5)
```

```
########## for theta = 0.5

for(n in seq(5,100 ,by = 5)){


  a= distance_comparision(n,0.5)
  distances[n/5,1] = a[1]
  distances[n/5,2]= a[2]

}
plot(seq(5,100 ,by = 5),distances[,1],xlab = "dimension of square matrix
    ",
     ylab = "KLD Distance",type="p",col="red",
     main = "Comaprision of distances for theta 0.5",
     cex.main = .6)

lines(seq(5,100 ,by = 5),distances[,2],col="green")

legend("topleft", legend=c("Actual distance", "Approximated distance"),
       col=c("red", "Green"), lty=c(1,1), cex=0.5)


##### for theta = 0.7
distances = matrix(0, nrow = 20, ncol = 2)

for(n in seq(5,100 ,by = 5)){


  a= distance_comparision(n,0.7)
  distances[n/5,1] = a[1]
  distances[n/5,2]= a[2]

}
plot(seq(5,100 ,by = 5),distances[,1],xlab = "dimension of square matrix
    ",
     ylab = "KLD Distance",type="p",col="black",
     main = "Comaprision of distances for theta 0.7",cex.main = .6)

lines(seq(5,100 ,by = 5),distances[,2],col="green")

legend("topleft", legend=c("Actual distance", "Approximated distance"),
       col=c("black", "Green"), lty=c(1,1), cex=0.5)

##### for theta = 0.9
distances = matrix(0, nrow = 20, ncol = 2)

for(n in seq(5,100 ,by = 5)){


  a= distance_comparision(n,0.9)
  distances[n/5,1] = a[1]
  distances[n/5,2]= a[2]

}
plot(seq(5,100 ,by = 5),distances[,1],xlab = "dimension of square matrix
    ",
     ylab = "KLD Distance",type="p",col="red",
     main = "Comaprision of distances for theta 0.9",cex.main = .6)
lines(seq(5,100 ,by = 5),distances[,2],col="green")

legend("topleft", legend=c("Actual distance", "Approximated distance"),
       col=c("red", "Green"), lty=c(1,1), cex=0.5)


##### for theta = 0.95
distances = matrix(0, nrow = 20, ncol = 2)

for(n in seq(5,100 ,by = 5)){


  a= distance_comparision(n,0.95)
  distances[n/5,1] = a[1]
  distances[n/5,2]= a[2]

}
plot(seq(5,100 ,by = 5),distances[,1],xlab = "dimension of square matrix
    ",
     ylab = "KLD Distance",type="p",col="red",
     main = "Comaprision of distances for theta 0.95",cex.main = .6)
lines(seq(5,100 ,by = 5),distances[,2],col="green")

legend("topleft", legend=c("Actual distance", "Approximated distance"),
       col=c("red", "Green"), lty=c(1,1), cex=0.5)









curve((30)/2*(1-exp(-30))*exp(-30*abs(x)),
      from = -1, to = 1,
      col = "blue", xlab = "Correlation Scale",
      ylab = "PC prior Point Denisity Function")
curve((10)/2*(1-exp(-10))*exp(-10*abs(x)),
      from = -1, to = 1,
      col = "red", xlab = "Correlation Scale",
      ylab = "PC prior Point Denisity Function", add = TRUE)

curve((2)/2*(1-exp(-2))*exp(-2*abs(x)),
      from = -1, to = 1,
      col = "black", xlab = "Correlation Scale",
      ylab = "PC prior Point Denisity Function", add = TRUE)


legend("topright", legend=c("lambda1 =2", "lambda1 =10","lambda1 =30"),
       col=c("black","red", "blue"), lty=c(1,1,1), cex=0.8)
title(main = "PC Prior for correlation for MA(1)")



### comaprision for varius n when fixing lambda =1

```

```
curve(exp(-sqrt(100*x^2-log(1-x^2)))*abs((100*x+(x/(1-x^2))))/(sqrt(100*
    x^2-log(1-x^2))),
       from = 0, to = 1,
       col = "blue",
       xlab = "Autocorelation Scale",ylab = "PC prior Point Denisity
           Function")

curve(exp(-sqrt(10*x^2-log(1-x^2)))*abs((10*x+(x/(1-x^2))))/(sqrt(10*x
    ^2-log(1-x^2))),
       from = 0, to = 1,
       col = "black",
       add=TRUE)

curve(exp(-sqrt(1*x^2-log(1-x^2)))*abs((1*x+(x/(1-x^2))))/(sqrt(1*x^2-
    log(1-x^2))),
       from = 0, to = 1,
       col = "red",
       add=TRUE)

curve(exp(-sqrt(1000*x^2-   log(1-x^2)))*abs((1000*x+(x/(1-x^2))))/(sqrt
    (1000*x^2-log(1-x^2))),
       from = 0, to = 1,
       col = "green",
       add=TRUE)

curve(exp(-sqrt(100000*x^2-log(1-x^2)))*abs((100000*x+(x/(1-x^2))))/(
    sqrt(100000*x^2-log(1-x^2))),
       from = 0, to = 1,
       col = "yellow",
       add=TRUE)


legend("topright", legend=c("n=102","n =12", "n =3","n=1002","n=100002")
    ,
        col=c("blue","black","red", "green","yellow"), lty=c(1,1,1,1,1),
            cex=0.8)
title(main = "PC Prior for autocorrelation for MA(1) process")
```

B Appendix A.2

Following code contains R code for comparing actual distance and approximated distance for ARMA(1,1) process, generates curves for Joint PC priors for ARMA(1,1) process.

```
library(fields);
library(geoR);
library(MASS)
#install.packages("akima")
library(akima)
library("spatial", lib.loc="~/R/x86_64-pc-linux-gnu-library/3.3")
#install.packages("matrixStats")
library("matrixStats")
```

```
distance_comparision = function(n,theta,  phi) {


  sigma_ar = matrix(0, nrow = n, ncol = n)
  sigma_ma = matrix(0, nrow = n, ncol = n)
  sigma_arma = matrix(0, nrow = n, ncol = n)
  sigma_arpprox = matrix(0, nrow = n, ncol = n)

  #sigma1_approx = matrix(0, nrow = n, ncol = n)
  for(i in 1:n) {

    for(j in 1:n) {

      if(abs(i-j) ==0)  {
        tmp = (1+theta^2+2*phi*theta)/(1-phi^2)
        tmp_ma = (1+theta^2)
        tmp_ar = 1/(1-phi^2)
      }
      else if (abs(i-j)==1) {
        tmp = (phi+theta)*(1+phi*theta)/(1-phi^2)
        tmp_ma = theta
        tmp_ar = phi*1/(1-phi^2)
      }

      else {
        tmp = (phi)^(abs(i-j)-1)*(phi+theta)*(1+phi*theta)/(1-phi^2)
        tmp_ma = 0
        tmp_ar =(phi)^(abs(i-j))*1/(1-phi^2)
      }

      sigma_arma[i,j] = tmp
      sigma_ma[i,j] = tmp_ma
      sigma_ar[i,j] = tmp_ar


    }

  }



  sigma_approx = sigma_ar %*% sigma_ma

  Distance_actual = (2*(abs(sum(diag(sigma_arma)) - n - log(det(sigma_
      arma)))))^.5
  Distance_approximated =(2*(abs(sum(diag(sigma_approx)) - n - log(det(
      sigma_approx)))))^.5

  return(c(Distance_actual,Distance_approximated,sum(diag(sigma_approx))
      ))

}

par(pty="s")
par(mfrow=c(1,3))

phi_axis = seq(from =-0.9, to=.9, by=.1 )
```

```
theta_axis = seq(from =-0.9, to=.9, by=.1 )


## For n =10
distance_grid_10 = matrix(0, nrow = 19, ncol = 19)
distance_grid_approx_10 =matrix(0, nrow = 19, ncol = 19)
#diag_sum = matrix(0, nrow = 20, ncol = 20)
distance_final_approximation_10 = matrix(0, nrow = 19, ncol = 19)

for(s in 1:19){
  for(p in 1:19) {

    n=10
    k= (s-10)/10
    l= (p-10)/10
    a= distance_comparision(n,k,l)
    #distances[n/5,1] = a[1]
    #distances[n/5,2]= a[2]
    distance_grid_10[s,p] = a[1]
    distance_grid_approx_10[s,p] = a[2]
    #diag_sum[s,p] = a[3]
    distance_final_approximation_10[s,p] = sqrt(9)*abs(k+l)

    #print(s,p)
  }
}
image.plot(phi_axis, theta_axis, distance_grid_10,xlab= expression(paste
    (phi)), ylab= expression(paste(theta)),main = "Actual distance ",
    cex=0.8)
contour(phi_axis, theta_axis, distance_grid_10, add = TRUE)
image.plot(phi_axis, theta_axis,distance_grid_approx_10, xlab=
    expression(paste(phi)), ylab= expression(paste(theta)),main = "1st
    level Approximated distance ", cex=0.8)
contour(phi_axis, theta_axis, distance_grid_approx_10, add = TRUE)

#distance_final_approximation = matrix(0, nrow = 20, ncol = 20)
#image.plot(diag_sum)
#image.plot(phi_axis, theta_axis,distance_final_approximation_10, main =
     "final Approximated distance ", cex=0.8)
image.plot(phi_axis, theta_axis, distance_grid_approx_10 - distance_grid
    _10,xlab= expression(paste(phi)), ylab= expression(paste(theta)),
    main = "Approximated distance -Actual distance", cex=0.8)
contour(phi_axis, theta_axis, distance_grid_approx_10 - distance_grid_
    10, add = TRUE)

## For n =50
distance_grid_50 = matrix(0, nrow = 19, ncol = 19)
distance_grid_approx_50 =matrix(0, nrow = 19, ncol = 19)
distance_final_approximation_50 = matrix(0, nrow = 19, ncol = 19)

for(s in 1:19){
  for(p in 1:19) {

    n=50
    k= (s-10)/10
    l= (p-10)/10
    a= distance_comparision(n,k,l)
```

```
      #distances[n/5,1] = a[1]
      #distances[n/5,2]= a[2]
      distance_grid_50[s,p] = a[1]
      distance_grid_approx_50[s,p] = a[2]
      distance_final_approximation_50[s,p] = sqrt(49)*abs(k+l)
      #print(s,p)
  }
}

image.plot(phi_axis, theta_axis, distance_grid_50,xlab= expression(paste
    (phi)), ylab= expression(paste(theta)),main = "Actual distance ",
    cex=0.8)
contour(phi_axis, theta_axis, distance_grid_50, add = TRUE)
image.plot(phi_axis, theta_axis,distance_grid_approx_50, xlab=
    expression(paste(phi)), ylab= expression(paste(theta)),main = "1st
    level Approximated distance ", cex=0.8)
contour(phi_axis, theta_axis, distance_grid_approx_50, add = TRUE)
#title(main = "Distance Comparision when n =50")

#distance_final_approximation = matrix(0, nrow = 20, ncol = 20)
#image.plot(diag_sum)
#image.plot(phi_axis, theta_axis,distance_final_approximation_10, main =
     "final Approximated distance ", cex=0.8)
image.plot(phi_axis, theta_axis, distance_grid_approx_50 - distance_grid
    _50,xlab= expression(paste(phi)), ylab= expression(paste(theta)),
    main = "Approximated distance -Actual distance", cex=0.8)
contour(phi_axis, theta_axis, distance_grid_approx_50 - distance_grid_
    50, add = TRUE)
#mtext("KLD Comparision for n =50", outer = FALSE, cex = 1)

## For n =100
distance_grid_100 = matrix(0, nrow = 19, ncol = 19)
distance_grid_approx_100 =matrix(0, nrow = 19, ncol = 19)
distance_final_approximation_100 = matrix(0, nrow = 19, ncol = 19)


for(s in 1:19){
  for(p in 1:19) {

      n=100
      k= (s-10)/10
      l= (p-10)/10
      a= distance_comparision(n,k,l)
      #distances[n/5,1] = a[1]
      #distances[n/5,2]= a[2]
      distance_grid_100[s,p] = a[1]
      distance_grid_approx_100[s,p] = a[2]
      distance_final_approximation_100[s,p] = sqrt(99)*abs(k+l)
      #print(s,p)
  }
}
image.plot(phi_axis, theta_axis, distance_grid_100,xlab= expression(
    paste(phi)), ylab= expression(paste(theta)),main = "Actual Distance
    ", cex=0.8)
contour(phi_axis, theta_axis, distance_grid_100, add = TRUE)
image.plot(phi_axis, theta_axis,distance_grid_approx_100, xlab=
    expression(paste(phi)), ylab= expression(paste(theta)),main = "1st
    level Approximated Distance ", cex=0.8)
contour(phi_axis, theta_axis, distance_grid_approx_100, add = TRUE)

#distance_final_approximation = matrix(0, nrow = 20, ncol = 20)
#image.plot(diag_sum)
#image.plot(phi_axis, theta_axis,distance_final_approximation_10, main =
     "final Approximated distance ", cex=0.8)
image.plot(phi_axis, theta_axis, distance_grid_approx_100 - distance_
    grid_100, xlab= expression(paste(phi)), ylab= expression(paste(theta
    )),main = "Approximated distance -Actual distance", cex=0.8)
contour(phi_axis, theta_axis, distance_grid_approx_100 - distance_grid_
    100, add = TRUE)
#mtext("KLD Comparision for n =100", outer = TRUE, cex = 1)

#image.plot(phi_axis, theta_axis, distance_final_approximation -
    distance_grid)


## For n =500
distance_grid_500 = matrix(0, nrow = 19, ncol = 19)
distance_grid_approx_500 =matrix(0, nrow = 19, ncol = 19)
distance_final_approximation_500 = matrix(0, nrow = 19, ncol = 19)


for(s in 1:19){
  for(p in 1:19) {

     n=500
     k= (s-10)/10
     l= (p-10)/10
     a= distance_comparision(n,k,l)
     #distances[n/5,1] = a[1]
     #distances[n/5,2]= a[2]
     distance_grid_500[s,p] = a[1]
     distance_grid_approx_500[s,p] = a[2]
     distance_final_approximation_500[s,p] = sqrt(99)*abs(k+l)
     #print(s,p)
  }
}
par(pty="s")
par(mfrow=c(1,3))

image.plot(phi_axis, theta_axis, distance_grid_500,xlab= expression(
    paste(phi)), ylab= expression(paste(theta)),main = "Actual distance
    ", cex=0.8)
contour(phi_axis, theta_axis, distance_grid_500, add = TRUE)
image.plot(phi_axis, theta_axis,distance_grid_approx_500, xlab=
    expression(paste(phi)), ylab= expression(paste(theta)),main = "1st
    level Approximated distance ", cex=0.8)
contour(phi_axis, theta_axis, distance_grid_approx_500, add = TRUE)

#distance_final_approximation = matrix(0, nrow = 20, ncol = 20)
#image.plot(diag_sum)
#image.plot(phi_axis, theta_axis,distance_final_approximation_10, main =
     "final Approximated distance ", cex=0.8)
image.plot(phi_axis, theta_axis, distance_grid_approx_500 - distance_
    grid_500,xlab= expression(paste(phi)), ylab= expression(paste(theta)
    ),main = "Approximated distance -Actual distance", cex=0.8)
contour(phi_axis, theta_axis, distance_grid_approx_500 - distance_grid_
    500, add = TRUE)
#mtext("KLD Comparision for n =100", outer = TRUE, cex = 1)


## For n =1000
distance_grid_1000 = matrix(0, nrow = 19, ncol = 19)
distance_grid_approx_1000 =matrix(0, nrow = 19, ncol = 19)
distance_final_approximation_1000 = matrix(0, nrow = 19, ncol = 19)


for(s in 1:19){
  for(p in 1:19) {

     n=1000
     k= (s-10)/10
     l= (p-10)/10
     a= distance_comparision(n,k,l)
     #distances[n/5,1] = a[1]
     #distances[n/5,2]= a[2]
     distance_grid_1000[s,p] = a[1]
     distance_grid_approx_1000[s,p] = a[2]
     distance_final_approximation_1000[s,p] = sqrt(99)*abs(k+l)
     #print(s,p)
  }
}
par(pty="s")
par(mfrow=c(1,3))

image.plot(phi_axis, theta_axis, distance_grid_1000,xlab= expression(
    paste(phi)), ylab= expression(paste(theta)),main = "Actual distance
    ", cex=0.8)
contour(phi_axis, theta_axis, distance_grid_1000, add = TRUE)
image.plot(phi_axis, theta_axis,distance_grid_approx_1000, xlab=
    expression(paste(phi)), ylab= expression(paste(theta)),main = "1st
    level Approximated distance ", cex=0.8)
contour(phi_axis, theta_axis, distance_grid_approx_1000, add = TRUE)

#distance_final_approximation = matrix(0, nrow = 20, ncol = 20)
#image.plot(diag_sum)
#image.plot(phi_axis, theta_axis,distance_final_approximation_10, main =
     "final Approximated distance ", cex=0.8)
image.plot(phi_axis, theta_axis, distance_grid_approx_1000 - distance_
    grid_1000,xlab= expression(paste(phi)), ylab= expression(paste(theta
    )),main = "Approximated distance -Actual distance", cex=0.8)
contour(phi_axis, theta_axis, distance_grid_approx_1000 - distance_grid_
    1000, add = TRUE)
#mtext("KLD Comparision for n =1000", outer = TRUE, cex = 1)




### Shape of ARMA(1,1) priors
phi_axis <- seq(from =-0.9, to=0.9, by=.1 )
theta_axis <- seq(from =-0.9, to=0.9, by=.1 )
```

```
### when lambda1 = 1

par(pty="s")
par(mfrow=c(1,1))
pc_prior_1 = matrix(0, nrow = 19, ncol = 19)

for(s in 1:19){
  for(p in 1:19) {

    k= (s-10)/10
    l= (p-10)/10
    pc_prior_1[s,p]=(1/12.78)*exp(-1*abs(k+l))/(2-abs(k+l))
  }
}

image.plot(phi_axis, theta_axis,pc_prior_1,main = "Joint PC priors for
   ARMA(1,1) ", cex=0.8)
persp(phi_axis,theta_axis,pc_prior_1, col = "BLUE",main = "Joint PC
    priors for ARMA(1,1)", cex=0.8)

### when lambda1 = 10

pc_prior_2 = matrix(0, nrow = 19, ncol = 19)

for(s in 1:19){
  for(p in 1:19) {

    k= (s-10)/10
    l= (p-10)/10
    pc_prior_2[s,p]=(1/12.78)*exp(-10*abs(k+l))/(2-abs(k+l))
  }
}

image.plot(phi_axis, theta_axis,pc_prior_2,main = "Joint PC priors for
   ARMA(1,1)", cex=0.8)
persp(phi_axis,theta_axis,pc_prior_2, col = "BLUE", main = "Joint PC
    priors for ARMA(1,1)", cex=0.8)


### when lambda1 = .1

pc_prior_3 = matrix(0, nrow = 19, ncol = 19)

for(s in 1:19){
  for(p in 1:19) {

    k= (s-10)/10
    l= (p-10)/10
    pc_prior_3[s,p]=(1/12.78)*exp(-.1*abs(k+l))/(2-abs(k+l))
  }
}

image.plot(phi_axis, theta_axis,pc_prior_3,main = "Joint PC priors for
   ARMA(1,1)", cex=0.8)
persp(phi_axis,theta_axis,pc_prior_3, col = "BLUE",main = "Joint PC
    priors for ARMA(1,1)", cex=0.8)
```

C Appendix A.3

R Code fit simulated AR(1) data using INLA framework, PC prior in this case are used considering the base model $\phi = 0$.

```
####Case 1 for different values of use defined
###scale when base model is considered with model rho =0

require(MASS)
require(sos)
library(INLA)
library(spam)
library(MASS)
n = 50
rho = 0.4
s = 0.1

#simulate a AR(1) data for a given marginal precision and rho

simulated_ts = scale(as.numeric(arima.sim(n = n, list(ar=c(rho)))))+
    rnorm(n, sd=s)

# define values for a suitable guess
u  = .4
alpha = c(0.5, 0.05 , 0.005)

# define prior distribution for hyper parameters
# define data frame as a input to inla
data = list(y=simulated_ts, z=1:n)

#for (i in 1:3) {

formula1 = simulated_ts~ -1 + f(z,model="ar1",
                                 hyper = list(rho = list(prior = "pc.cor0
                                     ",
                                                     param = c(u,
                                                         0.5)),
                                              prec = list(initial = log
                                                  (1), fixed=TRUE)))

formula2 = simulated_ts~ -1 + f(z,model="ar1",
                                 hyper = list(rho = list(prior = "pc.cor0
                                     ",
                                                     param = c(u,
                                                         0.05)),
                                              prec = list(initial = log
                                                  (1), fixed=TRUE)))


formula3 = simulated_ts~ -1 + f(z,model="ar1",
                                 hyper = list(rho = list(prior = "pc.cor0
                                     ",
                                                     param = c(u,
                                                         0.005)),
                                              prec = list(initial = log
                                                  (1), fixed=TRUE)))
```

```
r1 = inla (formula1, data = data,
           control.family = list(hyper = list(
             prec = list(initial = log(1/s^2), fixed = TRUE))))


r2 = inla (formula2, data = data,
           control.family = list(hyper = list(
             prec = list(initial = log(1/s^2), fixed = TRUE))))


r3 = inla (formula3, data = data,
           control.family = list(hyper = list(
             prec = list(initial = log(1/s^2), fixed = TRUE))))
r1=inla.hyperpar(r1)
r2=inla.hyperpar(r2)
r3=inla.hyperpar(r3)

plot(r1, plot.prior= T, single=TRUE)
plot(r2, plot.prior= T, single=TRUE)

plot(r3, plot.prior= T, single=TRUE)

r1$summary.hyperpar
r2$summary.hyperpar
r3$summary.hyperpar
```

D Appendix A.4

R Code fit simulated AR(1) time series data using INLA framework, PC prior in this cae are used considering the base model $\phi = 1$.

```
#Case 2 : for different values of use defined
###scale when base model is considered with model rho =1


n1 = 50
rho1 = 0.95
s1 = 0.1

#simulate a AR(1) data for a given marginal precision and rho

simulated_ts1 = scale(as.numeric(arima.sim(n = n1, list(ar=c(rho1)))))
                + rnorm(n, sd=s1)

# define values for a suitable guess
u1  = .95
alpha1 = c(0.3 ,0.7, 0.9)

# define prior distribution for hyper parameters
# define data frame as a input to inla
data1 = list(y=simulated_ts1, z=1:n1)
```

```
#for (i in 1:3) {


formula11 = simulated_ts1~ -1 + f(z,model="ar1",
                                       hyper = list(rho = list(prior = "pc.
                                           cor1",
                                                                  param = c(u1,
                                                                      alpha1[1])
                                                                      ),
                                                     prec = list(initial = log
                                                         (1), fixed=TRUE)))

formula12 = simulated_ts1~ -1 + f(z,model="ar1",
                                       hyper = list(rho = list(prior = "pc.
                                           cor1",
                                                                  param = c(u1,
                                                                      alpha1[2])
                                                                      ),
                                                     prec = list(initial = log
                                                         (1), fixed=TRUE)))
formula13= simulated_ts1~ -1 + f(z,model="ar1",
                                      hyper = list(rho = list(prior = "pc.
                                          cor1",
                                                                 param = c(u1,
                                                                     alpha1[3]))
                                                                     ,
                                                    prec = list(initial = log
                                                        (1), fixed=TRUE)))



r11 = inla (formula11, data = data1,
              control.family = list(hyper = list(
                prec = list(initial = log(1/s1^2), fixed = TRUE))))


r12 = inla (formula12, data = data1,
              control.family = list(hyper = list(
                prec = list(initial = log(1/s1^2), fixed = TRUE))))


r13 = inla (formula13, data = data1,
              control.family = list(hyper = list(
                prec = list(initial = log(1/s1^2), fixed = TRUE))))
#par(mfrow = c(3,2))
r11=inla.hyperpar(r11)
r12=inla.hyperpar(r12)
r13=inla.hyperpar(r13)

r11$summary.hyperpar
r12$summary.hyperpar
r13$summary.hyperpar


plot(r11, plot.prior= T, single=TRUE)
plot(r12, plot.prior= T, single=TRUE)
plot(r13, plot.prior= T, single=TRUE)
```

E Appendix A.5

R Code fit simulated MA(1) time series data using INLA framework,

```
require(MASS)
#require(sos)
library(INLA)
library(spam)


`inla.rgeneric.ma1.model` = function(
  cmd = c("graph", "Q", "mu", "initial", "log.norm.const", "log.prior",
      "quit"),
  theta = NULL, args = NULL)
{
  interpret.theta = function(n, theta, pars)
  {
    ## internal helper-function to map the parameters from the internal-
        scale to the
    ## user-scale
    return (list(prec = .01, # marginal precision for MA process,
                 rho = 2*exp(theta[1L])/(1+exp(theta[1L])) - 1.0))
    # unconstrained transformation for rho in theta
  }

  graph = function(n, theta, pars)
  {
    return (inla.as.sparse(matrix(1, n, n)))
  }

  Q = function(n, theta, pars)
  {
    par = interpret.theta(n, theta)
    S = toeplitz(c(1, par$rho/(1 + par$rho^2), rep(0, n-2)))
    S[1, 1] = S[n, n] = 1/(1 + par$rho^2)
    # S = S / p $ prev
    S = S / par$prec
    Q = solve(S)
    return (inla.as.sparse(Q))
  }

  mu = function(n, theta, pars)
  {
    return (numeric(0))
  }

  log.norm.const = function(n, theta, pars)
  {
    return(numeric(0))
  }

  log.prior = function(n, theta, pars)
  {
    ## return the log-prior for the hyperparameters. the '+theta[1L]' is
         the log(Jacobian)
    ## for having a gamma prior on the precision and convert it into the
         prior for the
    ## log(precision).
    param = interpret.theta(n, theta)
    #val = (dgamma(param$prec, shape = 1, rate = 1, log=TRUE) + theta[1L
       ] +
    #           dnorm(theta[2L], mean = 0, sd = 1, log=TRUE))
    if(is.null(pars)){ stop("stopped as pars are null")}

    fun_find_lambda = function(x) {return(exp(-x)*pars[2] +exp(-x*abs(
       pars[1])) - (pars[2]))}
    tt= uniroot(fun_find_lambda, c(0,1), extendInt = "yes")
    lambda = tt$root

    #lambda = -log(1-pars[2])/pars[1]

    #lambda = -log(0.05)
    val = log(2*lambda) - lambda* abs(param$rho) - theta[1] - 2*log(1+
       exp(-theta[1]))-log(2*(1-exp(-lambda)))
    return (val)
  }

  initial = function(n, theta, pars)
  {
    ## return initial values
    ntheta = 1
    return (rep(1, ntheta))
  }

  quit = function(n, theta, pars)
  {
    return (invisible())
  }

  cmd = match.arg(cmd)
  val = do.call(cmd, args = list(n = as.integer(args$n), theta = theta,
     pars = args$pars))
  return (val)
}


#simulate a MA(1) data for a given marginal precision and rho

n = 50
rho=0.4
s.MA = 10
x = arima.sim(n, model = list(ma = rho))*s.MA/sqrt(1+rho^2)
#x = scale(arima.sim(n, model = list(ma = rho)))


s = 0.01;
y = x + rnorm(n, sd = s);
```

```
# define prior distribution for hyper parameters
# define data frame as a input to inla



L =0.4
alpha =c(0.5, 0.05, 0.005);
#model = inla.rgeneric.define(inla.rgeneric.ma1.model, n=n)
model = inla.rgeneric.define(inla.rgeneric.ma1.model, n = n, pars = c
    (.4, .5));
formula = y ~ -1 + f(idx, model=model)
r1 = inla(formula, data = data.frame(y, idx = 1:n),
          family = "gaussian",
          control.family = list(hyper = list(prec = list(initial = log(1/
              s^2), fixed=TRUE))),
          verbose = TRUE)
summary(r1);
r11=inla.hyperpar(r1);
2/(1+exp(-r11$summary.hyperpar$mean))-1

plot(r11, plot.prior= T, single=TRUE)
xRange = range(r11$marginals.hyperpar[[1]][,1])
xs = seq(xRange[1], xRange[2], length.out = 100)
#xs = seq(-5, 5, length.out = 1000)
ys = xs
for(i in 1:length(xs)){
  ys[i] = inla.rgeneric.ma1.model(cmd ='log.prior', theta = xs[i], args
      = list(n = n, pars = c(.4, .5)))
}
lines(xs, exp(ys), col = "blue")

plot(inla.tmarginal(function(x){2/(1+exp(-x))-1},r11$marginals.hyperpar
    [[1]] ), type = "l",xlab = "Theta", main = "Posterior density for
    Theta in orignal scale" )
lines(inla.tmarginal(function(x){2/(1+exp(-x))-1},cbind(xs, exp(ys)) ),
    col = 'blue')
legend("topleft", legend=c("Posterior", "Prior"),
        col=c("black","blue"), lty=c(1,1), cex=0.8)

## for alpha = 0.05
model = inla.rgeneric.define(inla.rgeneric.ma1.model, n = n, pars = c
    (0.4, 0.05))
formula = y ~ -1 + f(idx, model=model)
r2 = inla(formula, data = data.frame(y, idx = 1:n),
          family = "gaussian",
          control.family = list(hyper = list(prec = list(initial = log(1/
              s^2), fixed=TRUE))))
summary(r2)
r12=inla.hyperpar(r2)
2/(1+exp(-r12$summary.hyperpar$mean))-1

plot(r12, plot.prior= T, single=TRUE)
xRange = range(r12$marginals.hyperpar[[1]][,1])
xs = seq(xRange[1], xRange[2], length.out = 100)
#xs = seq(-5, 5, length.out = 1000)
ys = xs
```

```
for(i in 1:length(xs)){
  ys[i] = inla.rgeneric.ma1.model(cmd ='log.prior', theta = xs[i], args
     = list(n = n, pars = c(0.4,0.05)))
}
lines(xs, exp(ys), col = "blue")

plot(inla.tmarginal(function(x){2/(1+exp(-x))-1},r12$marginals.hyperpar
    [[1]] ), type = "l",xlab = "Theta", main = "Posterior density for
    Theta in orignal scale" )
lines(inla.tmarginal(function(x){2/(1+exp(-x))-1},cbind(xs, exp(ys)) ),
    col = 'blue')
legend("topright", legend=c("Posterior", "Prior"),
       col=c("black","blue"), lty=c(1,1), cex=0.8)

## for alpha = 0.005
model = inla.rgeneric.define(inla.rgeneric.ma1.model, n = n, pars = c
    (0.4, 0.005))
formula = y ~ -1 + f(idx, model=model)
r3 = inla(formula, data = data.frame(y, idx = 1:n),
          family = "gaussian",
          control.family = list(hyper = list(prec = list(initial = log(1/
             s^2), fixed=TRUE))))
summary(r3)
r13=inla.hyperpar(r3)
2/(1+exp(-r13$summary.hyperpar$mean))-1

plot(r13, plot.prior= T, single=TRUE)
xRange = range(r13$marginals.hyperpar[[1]][,1])
xs = seq(xRange[1], xRange[2], length.out = 100)
#xs = seq(-5, 5, length.out = 1000)
ys = xs
for(i in 1:length(xs)){
  ys[i] = inla.rgeneric.ma1.model(cmd ='log.prior', theta = xs[i], args
     = list(n = n, pars = c(L, 0.005)))
}
lines(xs, exp(ys), col = "blue")

plot(inla.tmarginal(function(x){2/(1+exp(-x))-1},r13$marginals.hyperpar
    [[1]] ), type = "l",xlab = "Theta", main = "Posterior density for
    Theta in orignal scale" )
lines(inla.tmarginal(function(x){2/(1+exp(-x))-1},cbind(xs, exp(ys)) ),
    col = 'blue')
legend("topright", legend=c("Posterior", "Prior"),
       col=c("black","blue"), lty=c(1,1), cex=0.8)






#################### change of error varaince

n = 50
rho=0.4
s.MA = 10
x = arima.sim(n, model = list(ma = rho))*s.MA/sqrt(1+rho^2)
```

```
alpha = 0.5
s = c(0.01,0.5,1);
y1 = x + rnorm(n, sd = s[1]);
y2 = x + rnorm(n, sd = s[2]);
y3 = x + rnorm(n, sd = s[3]);

model = inla.rgeneric.define(inla.rgeneric.ma1.model, n = n, pars = c
    (.4, .5));
formula21 = y1 ~ -1 + f(idx, model=model)
r21 = inla(formula21, data = data.frame(y1, idx = 1:n),
           family = "gaussian",
           control.family = list(hyper = list(prec = list(initial = log(1
               /s[1]^2), fixed=TRUE))),
           verbose = TRUE)
summary(r21);
r21=inla.hyperpar(r21);
2/(1+exp(-r21$summary.hyperpar$mean))-1

plot(r21, plot.prior= T, single=TRUE)
xRange = range(r21$marginals.hyperpar[[1]][,1])
xs = seq(xRange[1], xRange[2], length.out = 100)
#xs = seq(-5, 5, length.out = 1000)
ys = xs
for(i in 1:length(xs)){
  ys[i] = inla.rgeneric.ma1.model(cmd ='log.prior', theta = xs[i], args
      = list(n = n, pars = c(.4, .5)))
}
lines(xs, exp(ys), col = "blue")

plot(inla.tmarginal(function(x){2/(1+exp(-x))-1},r21$marginals.hyperpar
    [[1]] ), type = "l",xlab = "Theta", main = "Posterior density for
    Theta in orignal scale" )
lines(inla.tmarginal(function(x){2/(1+exp(-x))-1},cbind(xs, exp(ys)) ),
    col = 'blue')
legend("topright", legend=c("Posterior", "Prior"),
       col=c("black","blue"), lty=c(1,1), cex=0.8)


######## when s = 0.5


formula22 = y2 ~ -1 + f(idx, model=model)
r22 = inla(formula22, data = data.frame(y2, idx = 1:n),
           family = "gaussian",
           control.family = list(hyper = list(prec = list(initial = log(1
               /s[2]^2), fixed=TRUE))),
           verbose = TRUE)
summary(r22);
r22=inla.hyperpar(r22);
2/(1+exp(-r22$summary.hyperpar$mean))-1

plot(r22, plot.prior= T, single=TRUE)
xRange = range(r22$marginals.hyperpar[[1]][,1])
xs = seq(xRange[1], xRange[2], length.out = 100)
#xs = seq(-5, 5, length.out = 1000)
ys = xs
for(i in 1:length(xs)){
  ys[i] = inla.rgeneric.ma1.model(cmd ='log.prior', theta = xs[i], args
     = list(n = n, pars = c(.4, .5)))
}
lines(xs, exp(ys), col = "blue")

plot(inla.tmarginal(function(x){2/(1+exp(-x))-1},r22$marginals.hyperpar
    [[1]] ), type = "l",xlab = "Theta", main = "Posterior density for
    Theta in orignal scale" )
lines(inla.tmarginal(function(x){2/(1+exp(-x))-1},cbind(xs, exp(ys)) ),
    col = 'blue')
legend("topright", legend=c("Posterior", "Prior"),
        col=c("black","blue"), lty=c(1,1), cex=0.8)



######### when s= 1


formula33 = y3 ~ -1 + f(idx, model=model)
r33 = inla(formula33, data = data.frame(y3, idx = 1:n),
            family = "gaussian",
            control.family = list(hyper = list(prec = list(initial = log
                (1/s[3]^2), fixed=TRUE))),
            verbose = TRUE)
summary(r33);
r33=inla.hyperpar(r33);
2/(1+exp(-r33$summary.hyperpar$mean))-1


plot(r33, plot.prior= T, single=TRUE)
xRange = range(r33$marginals.hyperpar[[1]][,1])
xs = seq(xRange[1], xRange[2], length.out = 100)
#xs = seq(-5, 5, length.out = 1000)
ys = xs
for(i in 1:length(xs)){
  ys[i] = inla.rgeneric.ma1.model(cmd ='log.prior', theta = xs[i], args
     = list(n = n, pars = c(.4, .5)))
}
lines(xs, exp(ys), col = "blue")

plot(inla.tmarginal(function(x){2/(1+exp(-x))-1},r33$marginals.hyperpar
    [[1]] ), type = "l",xlab = "Theta", main = "Posterior density for
    Theta in orignal scale" )
lines(inla.tmarginal(function(x){2/(1+exp(-x))-1},cbind(xs, exp(ys)) ),
    col = 'blue')
legend("topright", legend=c("Posterior", "Prior"),
        col=c("black","blue"), lty=c(1,1), cex=0.8)


par(mfrow=c(1,3))
```