Autoregressive model of order p (AR(p))

Parametrization

The autoregressive model of order p (AR1(p)) for the Gaussian vector $\mathbf{x} = (x_1, \ldots, x_n)$ is defined as (in obvious notation)

$$x_t = \phi_1 x_{t-1} + \phi_2 x_{t-2} + \cdots + \phi_p x_{t-p} + \epsilon_t$$

for $t = p, \ldots, n$, and where the innovation process $\{\epsilon_t\}$ has fixed precision.

The AR(p) process has an awkward parameterisation, as there are severe non-linear constraints on the ϕ.-parameters for it to define a stationary model. Therefore we re-parameterized using the partial autocorrelation autocorrelation function, $\{\psi_k, k = 1, \ldots, p\}$, where $|\psi_k| < 1$ for all k [1] and its *marginal (NOT conditional) precision* τ. Furthermore, the joint distribution for $\{x_t, t = 1, \ldots, p\}$, is set to the stationary distribution for the process, hence there are no boundary issues.

Hyperparameters

The marginal precision parameter τ is represented as

$$\theta_1 = \log(\tau)$$

and the prior for the marginal precision is defined on θ_1. The partial autocorrelation function $\{\psi_k\}$ is represented

$$\psi_k = 2\frac{\exp(\theta_{k+1})}{1 + \exp(\theta_{k+1})} - 1$$

for $k = 1, \ldots, p$. The prior for $\{\theta_{k+1}, k = 1, \ldots, p\}$ is *defined* to be multivariate normal with mean μ and precision matrix Q.

Specification

The AR(p) model is specified inside the `f()` function as

```
f(<whatever>, model="ar", order=<p>, hyper = <hyper>)
```

The option `order` (> 0) is required. The multivariate normal prior for $\{\theta_{k+1}, k = 1, \ldots, p\}$, `is specified` as the parameters to the prior for θ_2 (the first pacf-parameter), and the parameters to the multivariate normal prior (`mvnorm`), is `c(`μ, Q`)`; see the example below.

Hyperparameter spesification and default values

hyper

theta1

name log precision

short.name prec

initial 4

fixed FALSE

prior loggamma

param 1 5e-05

to.theta `function(x) log(x)`

from.theta `function(x) exp(x)`

[1] See for example `https://en.wikipedia.org/wiki/Partial_autocorrelation_function`. For $p = 1$, then $\psi_1 = \phi_1$, and for $p = 2$, then $\psi_1 = \phi_1/(1 - \phi_2)$ and $\psi_2 = \phi_2$.

theta2

name pacf1
short.name pacf1
initial 2
fixed FALSE
prior mvnorm
param 0 0.15
to.theta `function(x) log((1+x)/(1-x))`
from.theta `function(x) 2*exp(x)/(1+exp(x))-1`

theta3

name pacf2
short.name pacf2
initial 0
fixed FALSE
prior none
param
to.theta `function(x) log((1+x)/(1-x))`
from.theta `function(x) 2*exp(x)/(1+exp(x))-1`

theta4

name pacf3
short.name pacf3
initial 0
fixed FALSE
prior none
param
to.theta `function(x) log((1+x)/(1-x))`
from.theta `function(x) 2*exp(x)/(1+exp(x))-1`

theta5

name pacf4
short.name pacf4
initial 0
fixed FALSE
prior none
param
to.theta `function(x) log((1+x)/(1-x))`
from.theta `function(x) 2*exp(x)/(1+exp(x))-1`

theta6

name pacf5
short.name pacf5
initial 0
fixed FALSE
prior none
param
to.theta `function(x) log((1+x)/(1-x))`

from.theta `function(x) 2*exp(x)/(1+exp(x))-1`

theta7

- **name** pacf6
- **short.name** pacf6
- **initial** 0
- **fixed** FALSE
- **prior** none
- **param**
- **to.theta** `function(x) log((1+x)/(1-x))`
- **from.theta** `function(x) 2*exp(x)/(1+exp(x))-1`

theta8

- **name** pacf7
- **short.name** pacf7
- **initial** 0
- **fixed** FALSE
- **prior** none
- **param**
- **to.theta** `function(x) log((1+x)/(1-x))`
- **from.theta** `function(x) 2*exp(x)/(1+exp(x))-1`

theta9

- **name** pacf8
- **short.name** pacf8
- **initial** 0
- **fixed** FALSE
- **prior** none
- **param**
- **to.theta** `function(x) log((1+x)/(1-x))`
- **from.theta** `function(x) 2*exp(x)/(1+exp(x))-1`

theta10

- **name** pacf9
- **short.name** pacf9
- **initial** 0
- **fixed** FALSE
- **prior** none
- **param**
- **to.theta** `function(x) log((1+x)/(1-x))`
- **from.theta** `function(x) 2*exp(x)/(1+exp(x))-1`

theta11

- **name** pacf10
- **short.name** pacf10
- **initial** 0
- **fixed** FALSE
- **prior** none
- **param**

to.theta `function(x) log((1+x)/(1-x))`

from.theta `function(x) 2*exp(x)/(1+exp(x))-1`

constr FALSE

nrow.ncol FALSE

augmented FALSE

aug.factor 1

aug.constr

n.div.by

n.required FALSE

set.default.values FALSE

status experimental

pdf ar

Example

```
n = 100L
p = 2L
pacf = runif(p)
phi = inla.ar.pacf2phi(pacf)
y = arima.sim(n, model = list(ar = phi)) +
    rnorm(n, sd=sd(y)/100.0)
idx = 1L:n

param.prec = c(1, 0.01)
param.psi.mean = rep(0, p)
param.psi.prec = 0.15 * diag(p)
param.psi = c(param.psi.mean, param.psi.prec)

r = inla(y ~ -1 + f(
        idx, model='ar',
        order = p,
        hyper = list(
                ## marginal precision
                prec = list(param = param.prec),
                ## the parameters for the joint normal prior for the
                ## transformed pacf's, goes here.
                pacf1 = list(param = param.psi))),
        family = "gaussian",
        data = data.frame(y, idx))

## we will now estimate the posterior marginals of the phi-parameters
## using the (experimental) function 'inla.hyperpar.sampler', which
## creates samples from the approximated joint distribution for the
## hyperparameters.
nsamples = 100000
```

```
pacfs = inla.hyperpar.sampler(nsamples, r)[, 3L:(3L+(p-1L))]
phis = apply(pacfs, 1L, inla.ar.pacf2phi)
for(i in 1:p) {
    inla.dev.new()
    plot(density(phis[i, ]), main = paste("phi", i, sep=""))
    abline(v = phi[i])
}
```

Notes

- The functions `inla.ar.pacf2phi` and `inla.ar.phi2pacf` converts from the the ϕ-parameters to the ψ-parameters, using the Durbin-Levinson recursions. These can also be used to compute, the marginal posteriors of the ϕ-parameters from an approximation of the joint of the ϕ-parameters; see the example for a simulation based approach.
- Currently, the order p is limited to 10. If this creates a problem, let us know.
- If some of the ψ_k-parameters are fixed, and $k < p$, then the marginal (log-)likelihood is wrong; The joint normal prior for all the p ψ-parameters is used and not the conditional normal prior condition on the fixed ψ_k-parameters. If this creates a problem, let us know.
- The prior spesification for the multivariate normal is a bit awkward. Hopefully, we will come up with a better way to do this in the future.
- This model is currently marked as experimental.