



## MODELOS LINEARES DINÂMICOS DE DEFASAGEM POLINOMIAL COM LAG DISTRIBUÍDO

Luís Philipe Craveiro Mendes<sup>1</sup>

Rafael Santos Erbisti<sup>2</sup>

### Resumo

Modelos de defasagem polinomial com *lag* distribuído (DPLDM) têm sido utilizados em diversas áreas como finanças, economia, meio ambiente e engenharia. Em estudos ambientais, esses modelos são comumente aplicados para avaliar o impacto da concentração de poluentes e variáveis climáticas em desfechos da área da saúde. Geralmente, em séries temporais, percebe-se que o efeito de uma variável regressora não possui apenas o impacto imediato na resposta média do desfecho, mas que seus efeitos se propagam de alguma maneira para tempos futuros. Os DPLDM permitem essa interpretação imediata e defasada da influência da regressora sobre o desfecho, o que é particularmente útil para caracterizar o impacto total de efeitos persistentes ao longo do tempo. A especificação correta do modelo é essencial para identificar corretamente esses efeitos, especialmente em casos em que tipos sutis e não lineares de dependência temporal precisam ser identificadas. No contexto bayesiano, é possível incorporar defasagens polinomiais com *lag* distribuído nos Modelos Lineares Dinâmicos Normais (MLD). O procedimento de inferência é sequencial, permitindo a estimação instantânea dos parâmetros. A combinação desses modelos foi utilizada neste trabalho. Foram realizadas distintas simulações para diferentes cenários de efeitos defasados da variável regressora sobre o desfecho.

**Palavras-chave:** modelos dinâmicos gaussianos, modelos com *lag* distribuído, efeitos defasados, inferência bayesiana.

### Abstract

Polynomial distributed lag models (DPLDM) have been employed across various fields such as finance, economics, environmental science, and engineering. In environmental studies, these models are commonly utilized to assess the impact of pollutant concentration and climatic variables on health-related outcomes. Typically observed in time series, it is noted that the effect of a regressor variable not only has an immediate impact on the mean

---

<sup>1</sup> Instituto de Matemática e Estatística, Universidade Federal Fluminense. [luispcm@id.uff.br](mailto:luispcm@id.uff.br)

<sup>2</sup> Departamento de Estatística, Instituto de Matemática e Estatística, Universidade Federal Fluminense. [erbisti@id.uff.br](mailto:erbisti@id.uff.br)



response of the outcome but also propagates its effects in some manner to future time points. DPLDM enable this immediate and lagged interpretation of the regressor's influence on the outcome, which is particularly useful for characterizing the total impact of persistent effects over time. Correct model specification is essential for accurately identifying these effects, especially in cases where subtle and nonlinear types of temporal dependence need to be identified. In a Bayesian context, it is possible to incorporate polynomial lag-distributed delays into Normal Dynamic Linear Models (DLM). The inference procedure is sequential allowing for instantaneous parameter estimation. The combination of these models was utilized in this study. Simulations studies were conducted for different scenarios of lagged effects of the regressor variable on the outcome.

**Keywords:** lag model; dynamic linear model; lag effect; bayesian inference.

### Introdução

Modelos de defasagem polinomial com *lag* distribuído (DPLDM) têm sido amplamente aplicados na área ambiental (Gasparrini, 2016); (Schwartz, 2000). Uma grande vantagem de se utilizar os DPLDM é a possibilidade de entender o efeito imediato e defasado de uma variável regressora em um determinado desfecho, ou seja, podemos observar o efeito da variável regressora no momento do início de sua influência e ao passar do tempo.

Além disso, esses modelos podem identificar pequenas relações de dependência temporal não-lineares. Por exemplo, a diminuição de mortes por um determinada doença devido a algum fenômeno natural ou à ação humana. Recentemente, na literatura, estudos que visam analisar a relação entre poluentes e desfechos na saúde, como óbitos ou internações por doenças respiratórias, por meio de modelos com *lag* distribuído tem ganhado espaço (Belch et al., 2021); (Zhou et al., 2021). Considerando isso, os modelos polinomiais com *lag* distribuído não somente conseguem analisar a relação entre poluente e o desfecho, mas também a relação com variáveis climáticas que podem influenciar tal desfecho, além disso, os modelos possuem capacidade para capturar possíveis dependências não-lineares entre os poluentes e as variáveis climáticas e os desfechos.

Dentro do contexto bayesiano, procedimentos de inferência sequencial (West e Harrison, 1997) possibilitam a estruturação de Modelos Lineares Dinâmicos (MLD) que, por sua vez, suportam a discussão de modelos de lag distribuído com defasagem polinomial. Nesse sentido, aspectos envolvendo DPLDM foram abordados neste trabalho.

### Objetivo

Avaliar diferentes tipos de efeitos defasados de variáveis explicativas sobre a variável dependente, em estruturas de modelos dinâmicos bayesianos.

### Material e Método

Conforme (West e Harrison, 1997), considere uma série temporal  $Y = (Y_1, Y_2, \dots, Y_T)$ . O modelo linear dinâmico normal geral (MLD) é caracterizado pela quádrupla:  $\{F, G, V, W\}_t = \{F_t, G_t, V_t, W_t\}$ , para cada tempo  $t = 1, 2, \dots, T$ , o qual  $F_t$  é uma matriz conhecida de dimensão  $(n \times 1)$ ;  $G_t$  é uma matriz conhecida de dimensão  $(n \times n)$ ;  $V_t$  é a variância observacional de dimensão  $(1 \times 1)$ ; e  $W_t^*$  é uma matriz de variâncias conhecida de dimensão  $(n \times n)$  tal que  $W_t^* = (1 - \delta)G_t C_{t-1} G_t / \delta$ , sendo  $\delta$  o fator de desconto e  $D_0$  é a representação da informação inicial dada. Esta quádrupla define o modelo e relaciona  $Y_t$  ao parâmetro  $\theta_t$ , de dimensão  $(n \times 1)$  no tempo  $t$ . Assuma que a variância desconhecida é constante, ou seja,  $V_t = V$  para todo  $t$ , além disso considere a precisão observacional  $\phi = 1/V$ . As equações do modelo são definidas abaixo:

$$\text{Equação Observacional } Y_t = F_t' \theta_t + v_t, \quad v_t \sim N[0, V] \quad \text{Equação de Evolução } \theta_t = G_t \theta_{t-1} + \omega_t, \quad \omega_t \sim N[0, W_t^*]$$

As sequências de erros  $v_t$  e  $\omega_t$  são internamente e mutuamente independentes e são independentes de  $(\theta_0 | D_0)$ .

A ideia dos modelos de defasagem polinomial é uma variável dependente em função de uma outra variável de tal forma que a variável explicativa defasada pode explicar o comportamento da variável dependente de forma significativa. Além disso, uma vantagem importante é entender como a defasagem impacta a variável em cada período defasado. O modelo de defasagem polinomial é definido como:

$$Y_t = \alpha + \beta_0 X_t + \beta_1 X_{t-1} + \dots + \beta_q X_{t-q}.$$

A estimação dos coeficientes  $\beta_0, \beta_1, \dots, \beta_q$  retorna a informação sobre o padrão temporal do impacto de  $X$  com relação a  $q$  períodos atrás sobre a variável resposta,  $Y_t$ , no tempo  $t$ . Porém, a dependência que existe entre  $X_t$  e seus antecessores  $(X_{t-1}, X_{t-2}, \dots, X_{t-q})$  dificulta a estimação dos coeficientes devido a ocorrência de colinearidade entre  $(X_t, X_{t-1}, X_{t-2}, \dots, X_{t-q})$ . Para conseguirmos estimar os coeficientes, aproximamos a trajetória dos  $\beta_j$ 's, para  $j = 1, \dots, q$ , por meio de um polinômio de grau  $d \ll q$  da seguinte forma:

$\beta_j = \sum_{k=0}^d \eta_k j^k, j = 0, 1, \dots, q$ . A reparametrização anterior nos permite reescrever  $Y_t$  como segue:

$$Y_t = \alpha + \eta_0 S_{t,0} + \eta_1 S_{t,1} + \dots + \eta_d S_{t,d}.$$

ou seja, reduzimos o número de parâmetros que serão estimados de  $q$  para  $d$ , com

$S_{t,i} = \sum_{j=0}^q j^i X_{t-j}$ . Veja que o modelo anterior é mais parcimonioso, já que polinômios de baixo grau são suficientes para descrever curvas suaves ocasionadas pelos efeitos médios da defasagem de  $X$  em  $Y$ . Além disso,  $S_{t,i}$  não possui o problema de colinearidade,

possibilitando a estimação adequada dos  $\eta$ 's. Em geral, um grau  $d$  igual a 2 ou 3 já é suficiente para descrever os efeitos não lineares, porém a escolha de qual é melhor, num contexto aplicado, é feita com base em métricas de desempenho de modelos com ambos graus. Mais informações sobre a reparametrização podem ser achadas em (Almon, 1965).

É possível combinar o modelo de defasagem polinomial com os modelos lineares dinâmicos, os quais os estados  $\theta_t$  não têm evolução temporal. Essa combinação pode ser realizada da seguinte forma:

$$Y_t = F_t' \theta_t$$

onde  $\theta_t = \theta = (\alpha, \eta_0, \dots, \eta_d)$  e  $F_t' = (1, S_{t,0}, S_{t,1}, \dots, S_{t,d})$ . A partir desse modelo, podemos estimar os  $\eta$ 's conforme os procedimentos de inferência descritos em . Após a realização da estimação destes parâmetros, podemos reconstruir os coeficientes  $(\beta_0, \beta_1, \dots, \beta_q)$  nos permitindo identificar a influência média da defasagem da variável explicativa,  $X$ , para cada tempo defasado  $j$  na variável dependente,  $Y$ . Tal reconstrução se faz por meio da mesma relação definida anteriormente  $\beta_j = \sum_{k=0}^d \eta_k j^k, j = 0, 1, \dots, q$ . Para análise gráfica foi utilizado o pacote *tidyverse* (Wickham et al., 2019).

## Resultados e Discussão

O estudo de simulação busca representar diferentes tipos de impactos defasados de regressoras sobre o desfecho, a fim de avaliar a capacidade do modelo recuperar tais comportamentos. Em ambas as simulações, a variável explicativa foi simulada como:

$X_i = X_{i-1} + \epsilon, \epsilon \sim N[0, 10]$ , para  $i = 1, \dots, 200$ , onde  $X_0 = 8$ . Logo, a dimensão de  $Y$  é

igual a  $200 - q + 1$ , considerando  $q = 11$ , que é a defasagem utilizada para  $X$ , então  $dim(Y) = 190$ . Perceba que isso retorna  $\beta = (\beta_0, \dots, \beta_{10})$ , definindo a influência média da

defasagem de  $X$  em  $Y$ .

### Simulação 1

Considere o seguinte modelo para esta simulação, o qual foi utilizado um polinômio de grau  $d = 3$ :  $Y_t = F_t' \theta + v_t$ ,  $v_t \sim N[0, 5]$ , com  $F_t' = (1, S_{t,0}, S_{t,1}, S_{t,2}, S_{t,3})$  e  $\theta = (\alpha, \eta_0, \dots, \eta_3) = (20, 0.04, 1, -0.21, 0.011)$ . Além disso, as variâncias são fixas, bem como o fator de desconto é igual a 1, pois  $\theta$  é constante. A Figura 1 apresenta o resultado, o qual  $\beta$  fixado (verdadeiro) em rosa, e sua estimativa com intervalo de credibilidade de 95%. Note que o modelo consegue recuperar corretamente o comportamento do impacto de  $X$  em  $Y$ .

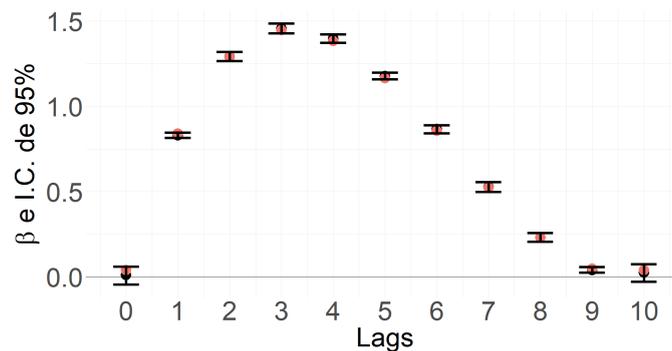


Figura 1.  $\beta$  estimado e verdadeiro: Simulação 1. Vermelho: valor verdadeiro. Preto: valor e i.c. estimados.

### Simulação 2

Já na simulação 2, o seguinte modelo foi utilizado:  $Y_t = F_t' \theta + v_t$ ,  $v_t \sim N[0, 5]$ , o qual  $F_t' = (1, S_{t,0}, S_{t,1}, S_{t,2})$  e  $\theta = (\alpha, \eta_0, \dots, \eta_2) = (20, 1.8, -0.38, 0.02)$ . A diferença entre esse modelo e o modelo da Simulação 1, é que  $\theta$  possui valores diferentes e  $d = 2$  o que acaba reduzindo a dimensão de  $F_t'$  e de  $\theta$ , já que  $d$  indica o grau do polinômio utilizado. E, novamente, o modelo conseguiu recuperar o comportamento do impacto ao longo do tempo.

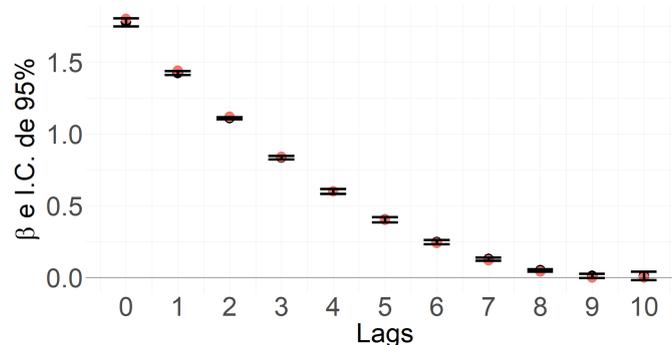


Figura 2.  $\beta$  estimado e verdadeiro: Simulação 2. Vermelho: valor verdadeiro. Preto: valor e i.c. estimados.



## Conclusão

O modelo proposto foi capaz de recuperar diferentes comportamentos de impactos defasados da regressora sobre o desfecho. A inferência bayesiana realizada na estimação dos parâmetros é instantânea e computacionalmente eficiente. Outros estudos simulados foram realizados com outros tipos de comportamento defasados das regressoras e, em todos, o modelo recuperou a estrutura gerada.

## Referências

ALMON, S. **The distributed lag between capital appropriations and expenditures**. *Econometrica*, JSTOR, v. 33, p. 178, JAN 1965. ISSN 0012-9682. Disponível em: (<https://dx.doi.org/10.2307/1911894>).

BELCH, J. J.; FITTON, C.; COX, B.; CHALMERS, J. D. **Associations between ambient air pollutants and hospital admissions: more needs to be done**. *Environmental Science and Pollution Research*, Springer Science and Business Media LLC, v. 28, p. 61848–61852, SEP 2021. ISSN 0944-1344, 1614-7499. Disponível em: (<https://dx.doi.org/10.1007/s11356-021-16544-0>).

GASPARRINI, A. **Modelling lagged associations in environmental time series data**. *Epidemiology, Ovid Technologies (Wolters Kluwer Health)*, v. 27, p. 835–842, NOV 2016. ISSN 1044-3983. Disponível em: (<https://dx.doi.org/10.1097/ede.0000000000000533>).

SCHWARTZ, J. **The distributed lag between air pollution and daily deaths**. *Epidemiology, Ovid Technologies (Wolters Kluwer Health)*, v. 11, p. 320–326, MAY 2000. ISSN 1044-3983. Disponível em: (<https://dx.doi.org/10.1097/00001648-200005000-00016>).

WEST, M.; HARRISON, J. **Bayesian Forecasting and Dynamic Models**. [S.l.]: Springer Science Business Media, 1997. ISBN 0387970258.

WICKHAM, H.; AVERICK, M.; BRYAN, J.; CHANG, W.; MCGOWAN, L. D.; FRANÇOIS, R.; GROLEMUND, G.; HAYES, A.; HENRY, L.; HESTER, J.; KUHN, M.; PEDERSEN, T. L.; MILLER, E.; BACHE, S. M.; MULLER, K.; OOMS, J.; ROBINSON, D.; SEIDEL, D. P.; SPINU, V.; TAKAHASHI, K.; VAUGHAN, D.; WILKE, C.; WOO, K.; YUTANI, H. **Welcome to the tidyverse**. *Journal of Open Source Software*, v. 4, n. 43, p. 1686, 2019.

ZHOU, H.; GENG, H.; DONG, C.; BAI, T. **The short-term harvesting effects of ambient particulate matter on mortality in taiyuan elderly residents: A time-series analysis with a generalized additive distributed lag model**. *Ecotoxicology and Environmental Safety*, Elsevier BV, v. 207, p. 111235, JAN 2021. ISSN 0147-6513. Disponível em: (<https://dx.doi.org/10.1016/j.ecoenv.2020.111235>).

## Anexo

```
library(tidyverse)
```

```
##### Gerando os dados #####  
set.seed(30)
```

```
x0 = 8
```



```
x=c()

alpha = 20

#variância do erro das regressoras:
tau2 = 10

#passo 1
x[1] = x0 + rnorm(1,0,sqrt(tau2))

n = 200 #quantidade de dados
for (i in 2:n){

  x[i] = x[i-1] + rnorm(1,0,sqrt(tau2))

}

#definindo os betas e o intercepto:

#vetor beta com dimensão
q = 11 #inclui o beta0

#montando a matriz de regressoras defasadas

X = matrix(nrow = n,
           ncol = q)

X[,1] = x #lag = 0

for (i in 2:q){

  X[,i] = dplyr::lag(x, i-1)

}

#X efetivo (sem os NA's)::

X = X[-c(1:10),]

##### Curva 1 #####

#vetor de etas verdadeiros:

eta_ver = c(0.04,1,-0.21,0.011)

M_aux = matrix(nrow = q-2,
               ncol = 4)

for (i in 2:q-1){

  M_aux[i-1,] = c(1, i, i^2, i^3)

}
```



```
beta_ver = c(eta_ver[1],
             sum(eta_ver),
             eta_ver%%M_aux[1,],
             eta_ver%%M_aux[2,],
             eta_ver%%M_aux[3,],
             eta_ver%%M_aux[4,],
             eta_ver%%M_aux[5,],
             eta_ver%%M_aux[6,],
             eta_ver%%M_aux[7,],
             eta_ver%%M_aux[8,],
             eta_ver%%M_aux[9,])

plot(x = 1:11, y = beta_ver)

#contruindo o St,i

#grau do polinômio d = 3
soma = 0

v = c(1,2,3,4,5,6,7,8,9,10) #vetor que multiplica cada matriz

St0 = apply(X,1,sum)
St1 = X[,-1]%%v^1
St2 = X[,-1]%%v^2
St3 = X[,-1]%%v^3

#gerando o Y

Y = c()

#variância do Y:
sigma2 = 5

Y = alpha + eta_ver[1]*St0 + eta_ver[2]*St1 + eta_ver[3]*St2 +
eta_ver[4]*St3 + rnorm(nrow(X), 0, sqrt(sigma2))

time = length(Y) #tamanho da série

##### Modelo Linear Dinâmico com Defasagem Polinomial #####

## ESTIMANDO A SERIE

n = 5          # Dimensão do vetor dos parâmetros a cada tempo

#vetor F

Ft = matrix(rep(1),n,time)

Ft[2,] = St0
Ft[3,] = St1
Ft[4,] = St2
Ft[5,] = St3
```



```
#matriz G
```

```
Gt = diag(rep(1,n))
```

```
Gt = array(Gt, dim = c(n,n,time))
```

```
##### Filtro de Kalman
```

```
### Passo t=0 ###
```

```
m0 = rep(10, n)
```

```
m0
```

```
C0 = diag(100,n,n)
```

```
C0
```

```
n0 = 2
```

```
S0 = 1
```

```
curve(dgamma(x,n0/2,S0/2), from =0 , to = 20)
```

```
##### Utilizando fator de desconto
```

```
delta = 1
```

```
D = 1/delta
```

```
##### Definindo as dimensões das matrizes e vetores
```

```
at = array(0, dim=c(n,1,time))
```

```
Rt = array(0, dim=c(n,n,time))
```

```
mt = array(0, dim=c(n,1,time))
```

```
Ct = array(0, dim=c(n,n,time))
```

```
ft = array(0, dim=c(1,1,time))
```

```
Qt = array(0, dim=c(1,1,time))
```

```
et = array(0, dim=c(1,1,time))
```

```
At = array(0, dim=c(n,1,time))
```

```
St = rep(0,time)
```

```
nt = rep(0,time)
```

```
### Passo t=1
```

```
# Priori em t=1
```

```
at[, ,1] = Gt[, ,1]**m0
```

```
Rt[, ,1] = (Gt[, ,1]**C0**(t(Gt[, ,1])))**D
```

```
# Previsão 1 passo-a-frente
```

```
ft[, ,1] = t(Ft[, ,1])**at[, ,1]
```

```
Qt[, ,1] = (t(Ft[, ,1])**Rt[, ,1]**Ft[, ,1]) + S0
```

```
# Posteriori em t=1
```

```
At[, ,1] = (Rt[, ,1]**Ft[, ,1])*(1/Qt[, ,1])
```

```
et[, ,1] = Y[1] - ft[, ,1]
```



```
#Equações de phi
nt[1] = n0 + 1
St[1] = S0 + (S0/nt[1])*(((et[,1])^2/Qt[,1])-1)

mt[,1] = at[,1] + (At[,1]*et[,1])
Ct[,1] = (St[1]/S0)*(Rt[,1] - At[,1]*%*t(At[,1])*Qt[,1])

for(t in 2:time){ # Passo 2 até o passo TIME

  # Priori em t
  at[,t] = Gt[,t]*%*mt[,t-1]
  Rt[,t] = (Gt[,t]*%*Ct[,t-1]*%*(t(Gt[,t])))*D

  # Previsão 1 passo-a-frente
  ft[,t] = t(Ft[,t])*%*at[,t]
  Qt[,t] = (t(Ft[,t])*%*Rt[,t]*%*Ft[,t]) + St[t-1]

  # Posteriori em t
  At[,t] = (Rt[,t]*%*Ft[,t])*%*(1/Qt[,t])
  et[,t] = Y[t] - ft[,t]

  #Equações de phi
  nt[t] = nt[t-1] + 1
  St[t] = St[t-1] + ((St[t-1]/nt[t])*(((et[,t])^2/Qt[,t])-1))

  mt[,t] = at[,t] + (At[,t]*et[,t])
  Ct[,t] = St[t]/St[t-1]*(Rt[,t] - At[,t]*%*t(At[,t])*Qt[,t])
}

##### Suavização #####

#definindo Bt:
Bt = array(0, dim=c(n,n,time))

mts = mt #isso porque para k = 0, o mts[time] = mt[time], ou seja, no
último tempo, mts e mt são iguais
Cts = Ct #o mesmo comentário de cima é válido para esse

for (k in 1:(time-1)){

  # k equivalente time-k          #Gt[,time-k+1]
  Bt[,time-k] = Ct[,time-k]*%*t(Gt[,time-k+1])*%*solve(Rt[,time+1-k])
  #no livro, at(-k) = mt no tempo t-k e Rt(-k) é Ct no mesmo tempo citado
  mts[,time-k] = mt[,time-k] + Bt[,time-k]*%*(mt[,time-k+1] -
at[,time-k+1])

  Cts[,time-k] = Ct[,time-k] +
Bt[,time-k]*%*(Ct[,time-k+1]-Rt[,time-k+1])*%*t(Bt[,time-k])
}

# Retornando para os betas e expondo o gráfico deles com intervalo de
confiança:
```



```
eta0 = mts[2,,length(Y)]#VEJA QUE USO OS VETORES SUAVIZADOS
eta1 = mts[3,,length(Y)]
eta2 = mts[4,,length(Y)]
eta3 = mts[5,,length(Y)]

#NOTE QUE TANTO PARA A ESPERANÇA DE ETA, MTS, E A COVARIÂNCIA DE ETA, CTS,
eu peguei a mais atualizada com base nas informações usando o argumento
length(Y)

covariancia_eta = Cts[-1,-1,length(Y)] #Cts[-1,-1] porque estamos tirando
as covariâncias que envolvem o intercepto.

eta = c(eta0,eta1,eta2,eta3) #vetor de etas

#matriz auxiliar
m = matrix(1, nrow = q,
           ncol = length(eta))

m[1,] = c(1,0,0,0)

for (i in 2:length(eta)){
  for (j in 1:q-1){

    m[j+1,i] = j^(i-1)

  }
}

betas_est = m%%eta #voltado para os betas

#Variância dos betas:
v_aux = 0

v_beta0 = as.numeric(covariancia_eta[1,1])
var = v_beta0

for (i in 1:(q-1)){

  v_aux = c(1, i, i^2, i^3)

  var = c(t(v_aux)%%covariancia_eta%%v_aux,
          var)

}

var = rev(var) #invertendo, pois o vetor estava com beta0 no final, como
penúltimo o beta1, assim em diante, ou seja, tava ao contrário do que
deveria ser

##### Intervalo de Credibilidade de 95% #####

#gerando amostras:
var = diag(var)
```



```
amostra_betas = rmvnorm(1000, betas_est, var)

ic_inf = apply(amostra_betas, MARGIN = 2, FUN = quantile, probs = 0.025)

ic_sup = apply(amostra_betas, MARGIN = 2, FUN = quantile, probs = 0.975)

##### Gráfico dos betas estimados #####
ggplot(NULL,
  aes(x = 0:10)) +
  geom_point(aes(y = betas_est), size = 8) +
  geom_point(aes(y = beta_ver, colour = "red"), size = 8, show.legend = F)
+
  geom_errorbar(aes(ymin = ic_inf, ymax = ic_sup),
    width = 0.5, size = 2) +
  geom_hline(yintercept = 0) +
  labs(x = "Lags",
    y = expression(paste(beta, " e I.C. de 95%")),
    colour = "") +
  theme_minimal() +
  theme(axis.title = element_text(size = 45),
    axis.text = element_text(size = 45),
    legend.text=element_text(size= 40),
    legend.title = element_text(size = 45),
    legend.position = NULL) +
  scale_x_continuous(breaks = seq(0, 10, by = 1))

##### Curva 2 #####

#vetor de etas verdadeiros:

eta_ver = c(1.8,-0.38,0.02)

M_aux = matrix(nrow = q-2,
  ncol = 3)

for (i in 2:q-1){
  M_aux[i-1,] = c(1, i, i^2)
}

beta_ver = c(eta_ver[1],
  sum(eta_ver),
  eta_ver%%M_aux[1,],
  eta_ver%%M_aux[2,],
  eta_ver%%M_aux[3,],
  eta_ver%%M_aux[4,],
  eta_ver%%M_aux[5,],
  eta_ver%%M_aux[6,],
  eta_ver%%M_aux[7,],
  eta_ver%%M_aux[8,],
  eta_ver%%M_aux[9,])
```



```
plot(x = 1:11, y = beta_ver)

#contruindo o St,i

#grau do polinômio d = 3
soma = 0

v = c(1,2,3,4,5,6,7,8,9,10) #vetor que multiplica cada matriz

St0 = apply(X,1,sum)
St1 = X[,-1]%%v^1
St2 = X[,-1]%%v^2
#St3 = X[,-1]%%v^3

#gerando o Y

Y = c()

#variância do Y:
sigma2 = 5

Y = alpha + eta_ver[1]*St0 + eta_ver[2]*St1 + eta_ver[3]*St2 +
rnorm(nrow(X), 0, sqrt(sigma2))

time = length(Y) #tamanho da série

##### Modelo Linear Dinâmico com Defasagem Polinomial #####

## ESTIMANDO A SERIE

n = 4          # Dimensão do vetor dos parâmetros a cada tempo (incluindo o
intercepto)

#vetor F

Ft = matrix(rep(1),n,time)

Ft[2,] = St0
Ft[3,] = St1
Ft[4,] = St2
#Ft[5,] = St3

#matriz G

Gt = diag(rep(1,n))

Gt = array(Gt, dim = c(n,n,time))

##### Filtro de Kalman

### Passo t=0 ###
m0 = rep(10, n)
m0
C0 = diag(100,n,n)
```



```
C0
n0 = 2
S0 = 1

curve(dgamma(x,n0/2,S0/2), from =0 , to = 20)

##### Utilizando fator de desconto

delta = 1
D = 1/delta

##### Definindo as dimensões das matrizes e vetores

at = array(0, dim=c(n,1,time))
Rt = array(0, dim=c(n,n,time))

mt = array(0, dim=c(n,1,time))
Ct = array(0, dim=c(n,n,time))

ft = array(0, dim=c(1,1,time))
Qt = array(0, dim=c(1,1,time))

et = array(0, dim=c(1,1,time))
At = array(0, dim=c(n,1,time))

St = rep(0,time)
nt = rep(0,time)

### Passo t=1

# Priori em t=1
at[,,1] = Gt[,,1]**m0
Rt[,,1] = (Gt[,,1]**C0**(t(Gt[,,1])))*D

# Previsão 1 passo-a-frente
ft[,,1] = t(Ft[,,1])**at[,,1]
Qt[,,1] = (t(Ft[,,1])**Rt[,,1]**Ft[,,1]) + S0

# Posteriori em t=1
At[,,1] = (Rt[,,1]**Ft[,,1])*(1/Qt[,,1])
et[,,1] = Y[1] - ft[,,1]

#Equações de phi
nt[1] = n0 + 1
St[1] = S0 + (S0/nt[1])*(((et[,,1])^2/Qt[,,1])-1)

mt[,,1] = at[,,1] + (At[,,1]*et[,,1])
Ct[,,1] = (St[1]/S0)*(Rt[,,1] - At[,,1]**t(At[,,1])*Qt[,,1])

for(t in 2:time){ # Passo 2 até o passo TIME

  # Priori em t
  at[,,t] = Gt[,,t]**mt[,,t-1]
```



```
Rt[, ,t] = (Gt[, ,t]**Ct[, ,t-1]**(t(Gt[, ,t])))**D

# Previsão 1 passo-a-frente
ft[, ,t] = t(Ft[, ,t])**at[, ,t]
Qt[, ,t] = (t(Ft[, ,t])**Rt[, ,t]**Ft[, ,t]) + St[t-1]

# Posteriori em t
At[, ,t] = (Rt[, ,t]**Ft[, ,t])**((1/Qt[, ,t]))
et[, ,t] = Y[t] - ft[, ,t]

#Equações de phi
nt[t] = nt[t-1] + 1
St[t] = St[t-1] + ((St[t-1]/nt[t])**((et[, ,t]^2/Qt[, ,t])-1))

mt[, ,t] = at[, ,t] + (At[, ,t]*et[, ,t])
Ct[, ,t] = St[t]/St[t-1]*(Rt[, ,t] - At[, ,t]**t(At[, ,t])*Qt[, ,t])

}

##### Suavização #####

#definindo Bt:
Bt = array(0, dim=c(n,n,time))

mts = mt #isso porque para k = 0, o mts[time] = mt[time], ou seja, no
último tempo, mts e mt são iguais
Cts = Ct #o mesmo comentário de cima é válido para esse

for (k in 1:(time-1)){

  # k equivalente time-k          #Gt[, ,time-k+1]
  Bt[, ,time-k] = Ct[, ,time-k]**t(Gt[, ,time-k+1])**solve(Rt[, ,time+1-k])
  #no livro, at(-k) = mt no tempo t-k e Rt(-k) é Ct no mesmo tempo citado
  mts[, ,time-k] = mt[, ,time-k] + Bt[, ,time-k]**(mt[, ,time-k+1] -
at[, ,time-k+1])

  Cts[, ,time-k] = Ct[, ,time-k] +
Bt[, ,time-k]**(Ct[, ,time-k+1]-Rt[, ,time-k+1])**t(Bt[, ,time-k])
}

# Retornando para os betas e expondo o gráfico deles com intervalo de
confiança:

eta0 = mts[2, ,length(Y)]#VEJA QUE USO OS VETORES SUAVIZADOS
eta1 = mts[3, ,length(Y)]
eta2 = mts[4, ,length(Y)]
#eta3 = mts[5, ,length(Y)]

#NOTE QUE TANTO PARA A ESPERANÇA DE ETA, MTS, E A COVARIÂNCIA DE ETA, CTS,
eu peguei a mais atualizada com base nas informações usando o argumento
length(Y)

covariancia_eta = Cts[-1,-1,length(Y)] #Cts[-1,-1] porque estamos tirando
as covariâncias que envolvem o intercepto.
```



```
eta = c(eta0,eta1,eta2)#eta3) #vetor de etas

#matriz auxiliar
m = matrix(1, nrow = q,
           ncol = length(eta))

m[1,] = c(1,0,0)#0

for (i in 2:length(eta)){
  for (j in 1:q-1){

    m[j+1,i] = j^(i-1)

  }
}

betas_est = m%%eta #voltado para os betas

#Variância dos betas:
v_aux = 0

v_beta0 = as.numeric(covariancia_eta[1,1])
var = v_beta0

for (i in 1:(q-1)){

  v_aux = c(1, i, i^2) #i^3

  var = c(t(v_aux)%%covariancia_eta%%v_aux,
          var)

}

var = rev(var) #invertendo, pois o vetor estava com beta0 no final, como
penúltimo o beta1, assim em diante, ou seja, tava ao contrário do que
deveria ser

##### Intervalo de Credibilidade de 95% #####

#gerando amostras:
var = diag(var)

amostra_betas = rmvnorm(1000, betas_est, var)

ic_inf = apply(amostra_betas, MARGIN = 2, FUN = quantile, probs = 0.025)
ic_sup = apply(amostra_betas, MARGIN = 2, FUN = quantile, probs = 0.975)

##### Gráfico dos betas estimados #####
ggplot(NULL,
       aes(x = 0:10)) +
  geom_point(aes(y = betas_est), size = 8) +
  geom_point(aes(y = beta_ver, colour = "red"), size = 8, show.legend = F)
```



```
+  
  geom_errorbar(aes(ymin = ic_inf, ymax = ic_sup),  
                width = 0.5, size = 2) +  
  geom_hline(yintercept = 0) +  
  labs(x = "Lags",  
        y = expression(paste(beta, " e I.C. de 95%")),  
        colour = "") +  
  theme_minimal() +  
  theme(axis.title = element_text(size = 45),  
        axis.text = element_text(size = 45),  
        legend.text=element_text(size= 40),  
        legend.title = element_text(size = 45),  
        legend.position = NULL) +  
  scale_x_continuous(breaks = seq(0, 10, by = 1))
```