Introdução

Existem duas formas básicas de estruturar um modelo: pela abordagem fenomenológica ou pela abordagem emp?rica. Na primeira há exigência de um conhecimento pr?vio da natureza de um sistema, na segunda os modelos s?o constru?dos paralelamente ao experimento, sem exigir um conhecimento da ess?ncia do sistema.

Do modo geral, os modelos podem ser classificados como est?ticos ou din?micos, discretos ou cont?nuos, determin?sticos ou estoc?sticos, param?tricos ou n?o param?tricos, lineares ou n?o lineares. Conforme a classifica??o, que n?o s?o mutuamente excludentes, t?m-se as ferramentas adequadas para o estudo do sistema.

No estudo de sistemas lineares duas abordagem se destacam: a da representa??o do sistema por fun??es de transfer?ncias e pela representa??o no espa?os de estados. A fun??es de transfer?ncias caracterizam por descrever a din?mica de um sistema mediante a rela??o das vari?veis de entradas com as de sa?da. J? a representa??o no espa?o de estados permite conhecer internamente o sistema, visto que utiliza-se das vari?veis internas para modelar o problema.

Algumas hip?teses s?o recorrentemente estabelecidas ao se identificar um sistema, visto que simplificam o tratamento do sistema, a saber, linearidade, invari?ncia no tempo e concentra??o de par?metros.

Neste trabalho estuda-se um sistema linear estoc?stico e est?tico, que apesar de cont?nuo, ser? tratado discretamente, utilizando filtro de Kalman para a estima??o da varia??o de posi??o perante um sistema com erro de medidas de posi??o.

Fórmulas

$x_{k+1} = Ax_k + Bu_k + w_k$

$y_k = Cx_k + z_k$

$x_{k+1} = \begin{bmatrix} 1 & t \\ 0 & 1 \end{bmatrix} x_k + \begin{bmatrix} \frac{t^2}{2} & t \end{bmatrix} u_k + w_k$

$y_k = \begin{bmatrix} 1 & 0 \end{bmatrix} x_k + z_k$

```
plotGraphs<-function(ideal,estimado,medido,sk,innt){

  t <- 1:length(ideal)

  aa <- (estimado-medido)^2/medido
  xx <- sum(aa)


  par(mfcol=c(3,1))

  plot(medido,pch=".",col="RED")
  points(ideal,pch=".")
  points(estimado,pch=".",col="GREEN",type="l")
  legend("bottomright",legend=c('Posição Medida','Posição idealizada','Posição Estimada'),fil=c("RED","

  plot(t,aa,pch=".",type="l",col="RED");
  points(t,-sqrt(sk)^2,pch=".",col="BLUE")
  points(t,sqrt(sk)^2,pch=".",col="GREEN")
  legend("bottomright",legend=c('Erros de Estimação(Inovação)','Limite Superior do Intervalo de Confian

  plot(t, innt^2,pch=".",type="l")
}
```

```
simular <- function(n,error,seed=12345){

  set.seed(seed)
  n <- n                           # Número de observações

  u <- 1                                # Aceleração idealizada para o experimento

  dt <- 0.1                             # Menor intervalo tempo medido no simulador

  measnoise <- 10                       # Erro de medida da posição
  accelnoise <- error                   # Erro da aceleração

  a <- matrix(data=c(1, 0, dt, 1), ncol=2,nrow=2)
  b <- matrix(data=c(dt^2/2, dt), nrow=2, ncol=1)
  c <- matrix(data=c(1,1),nrow=1,ncol=2)
  x <- matrix(data=c(0,0),nrow=2,ncol=1)# Posição inicial
  xhat <- matrix(data=c(14,14),nrow=2,ncol=1)                             # Posição estimada inicial
  Sz <- measnoise^2                     # Variância do ruído da medida
  Sw <- accelnoise^2*matrix(data=c(dt^4, dt^3/2, dt^3/2, dt^2),
                         nrow=2, ncol=2) # Estimativa
  P <- matrix(data=c(15,15,15,15),nrow=2,ncol=2)      # Matriz de covariância de estado (pode ser com qu
  Pr <- array(dim=c(n,2,2))
  pos <- array(dim=n)                    # Posição idealizada (durante todo o experimento)
  pos[1] <-0
  poshat <- array(dim=n)                  # Posição estimada (durante todo o experimento)
  poshat[1] <- 0
  posmeas <- array(dim=n)                # Posição medida (durante todo o experimento)
  posmeas[1] <- 0
  vel <- array(dim=n)                    # Velocidade idealizada (durante todo o experimento)
  vel[1] <- 0
  velhat <- array(dim=n)                 # Velocidade estimada (durante todo o experimento)
  velhat[1] <- 0
  zz <- array(dim=n)                     # Posição + ruído (durante todo o experimento)
  zz[1] <- 0

  Sk <- array(dim=n)
  qk <- array(dim=n)
  Innt <- array(dim=n)

  for(t in 1:n){
    ProcessNoise <- accelnoise * matrix(data=c(dt^2/2*rnorm(n=1), dt*rnorm(n=1)),
                                        nrow=2, ncol=1) # Ruído
    x <- a%*%x + b*u + ProcessNoise   # Posição "correta" + ruído
    MeasNoise <- measnoise * rnorm(n=1,sd = accelnoise) # Ruído da medição
    y <- c%*%x + MeasNoise              # Medição do ruído
    xhat <- a%*%xhat + b%*%u            # Estimação da posição
    Inn <- y - c%*%xhat                 # Matriz erro (innovation)

    s <- c%*%P%*%t(c) + Sz

    K <- a%*%P%*%t(c)%*%solve(s)        # Ganho

    xhat <- xhat + K%*%Inn              # Correção da estimação
```

```
    Pr[t,,] <- P
    P <- a%*%P%*%t(a) - a%*%P%*%t(c)%*%solve(s)%*%c%*%P%*%t(a) +
      Sw # Matriz de covariância (grau de incerteza das estimativas)

    Sk[t] <- c%*%P%*%t(c)+Sz

    Innt[t] <- Inn

    qk[t] <- Inn%*%solve(c%*%P%*%t(c) +Sz)%*%Inn

    pos[t] <- x[1]
    posmeas[t] <- y
    poshat[t] <- xhat[1]
    vel[t] <- x[2]
    velhat[t] <- xhat[2]

    z <- a%*%x + b%*%u # Posição sem ruído (idealizada)
    zz[t] <- z[1]
  }

  poshatComp<-chisq.test(matrix(data=c(pos-min(posmeas),poshat-min(posmeas)),nrow=n,ncol=2))


  posmeasComp<-chisq.test(matrix(data=c(posmeas-min(posmeas),pos-min(posmeas)),nrow=n,ncol=2))
  pose <- data.frame(pos=pos,poshat=poshat,posmeas=posmeas,sk=Sk,innt=Innt)
  return <- c(poshatComp,posmeasComp,pose)
}
```

Podemos fazer consultas sobres os dados retornados pela simulação. Percebam que não foram usados todos (olhando os índices da variável simulado), os outros dados são dados do teste χ^2 criados a longo do teste em si. Não creio que sirvam de muita coisa, então deixei de fora.

Os valores recuperados podem ser exibidos em chunks (com formatação padrão), ou *inline*, onde podemos aplicar formatação Markdown.

Notem que os valores recuperados e usados *inline* estão, em sua maioria, entre $, isto devido problemas que podem ocorrer com números expressos em notação científica, que o Latex sabe trabalhar, mas o Markdown, não.

Simulação para n=200

Segue abaixo modelo de simulações

Para este exemplo foi realizada simulação usando intervalo de tempo de 20 segundos.

O método usado foi: Pearson's Chi-squared test.

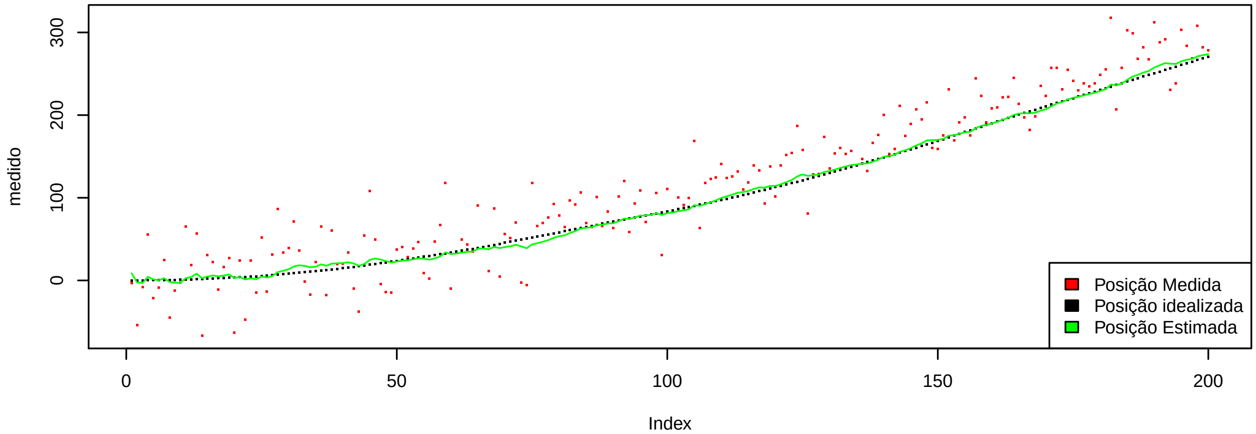
Para comparação entre a posição ideal e a estimada foram obtidos os seguinte valores:

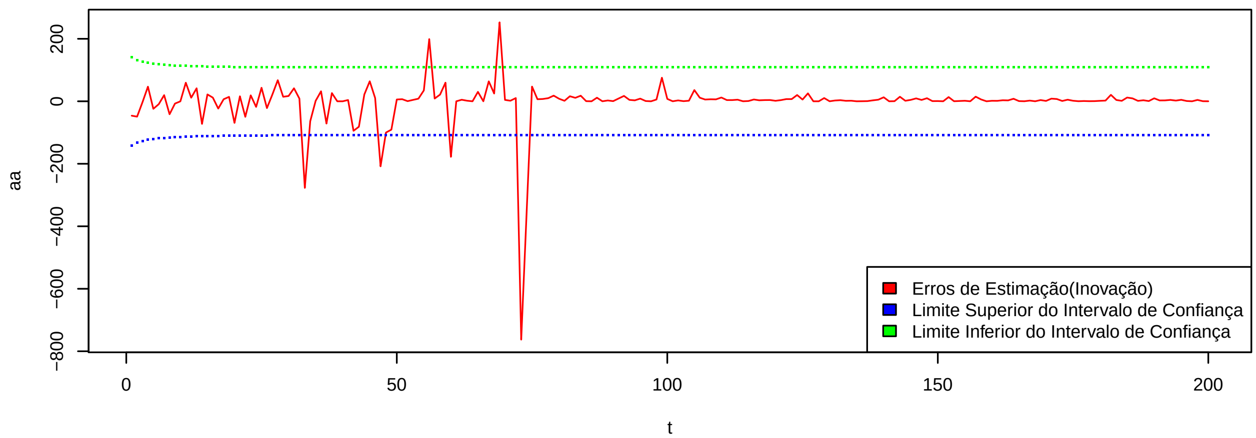
- χ^2: 11.3858295
- υ: 199
- Probabilidade de Semelhança: 1

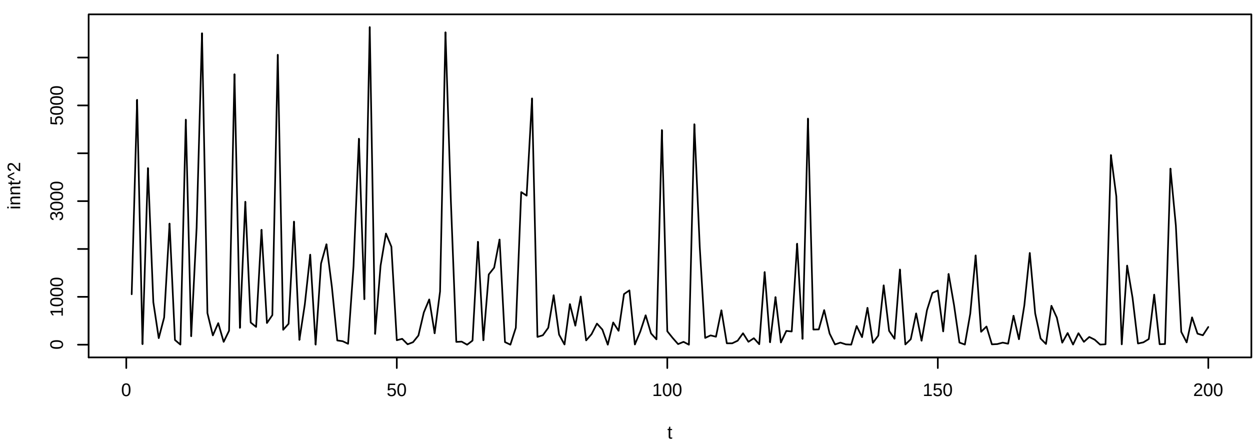
Para comparação entre a posição ideal e a medida foram obtidos os seguinte valores:

- χ^2: 792.6621503
- υ: 199
- Probabilidade de Semelhança: $8.6357302 \times 10^{-72}$

Plotando os gráficos da simulação







Simulação para n=1000

Para este exemplo foi realizada simulação usando intervalo de tempo de 100 segundos.

O método usado foi: Pearson's Chi-squared test.

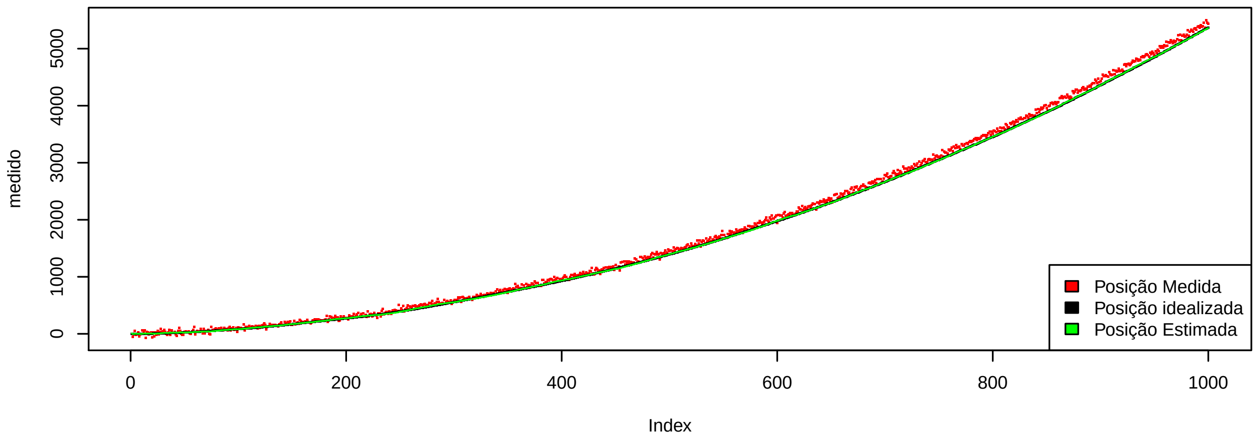
Para comparação entre a posição ideal e a estimada foram obtidos os seguinte valores:

- χ^2: 22.5509214
- υ: 999
- Probabilidade de Semelhança: 1

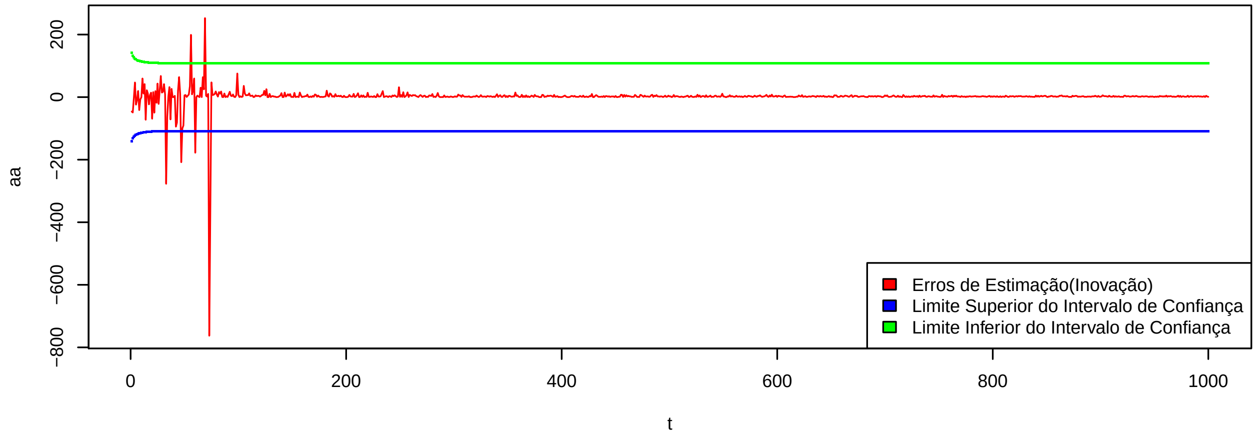
Para comparação entre a posição ideal e a medida foram obtidos os seguinte valores:

- χ^2: 1154.3691175
- υ: 999
- Probabilidade de Semelhança: 4.4177826×10^{-4}

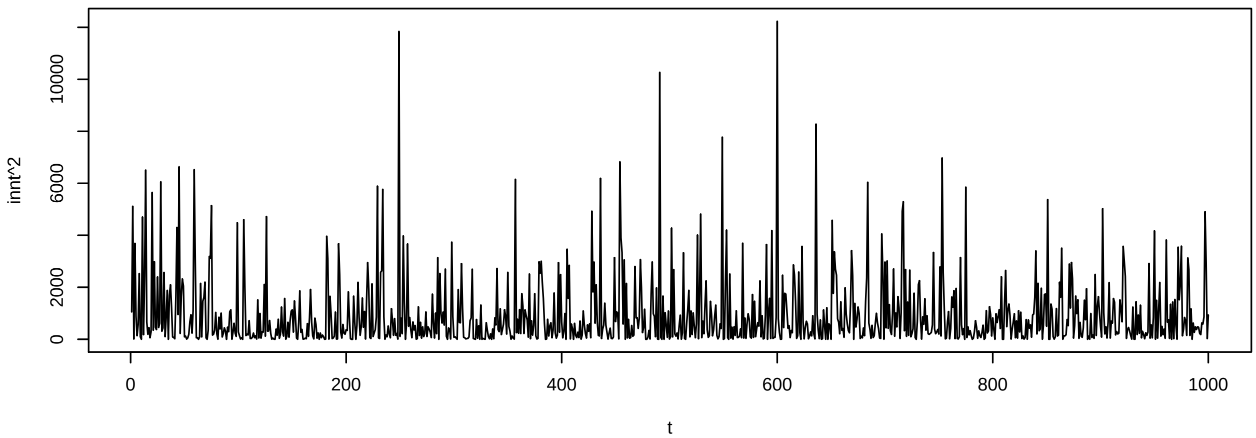
Plotando os gráficos da simulação



medido
0
1000
2000
3000
4000
5000
0
200
400
600
800
1000
Index
Posição Medida
Posição idealizada
Posição Estimada



aa
−800
−600
−400
−200
0
200
0
200
400
600
800
1000
t
Erros de Estimação(Inovação)
Limite Superior do Intervalo de Confiança
Limite Inferior do Intervalo de Confiança



innt^2
0
2000
6000
10000
0
200
400
600
800
1000
t