

A linha dos quadrados mínimos da diferença pode ser encontrada minimizando a soma

$$d_1^2 + d_2^2 + \dots + d_n^2 = \sum_{i=1}^n d_i^2.$$

Estas são as diferenças.

Dada a equação da linha do modelo $\hat{y} = b x + a$, em que b é o slope e a o deslocamento vertical.

Os valores na linha do modelo são denotados \hat{y} .

Cada distância vertical é a diferença entre \hat{y} e y .

Substituindo \hat{y} em

$$d_i = \hat{y}_i - y_i$$

por $b x + a$, temos

$$d_i = b x_i + a - y_i.$$

Com isto, a soma da minimização dos quadrados da diferença fica

$$(b x_1 + a - y_1)^2 + (b x_2 + a - y_2)^2 + \dots + (b x_n + a - y_n)^2 = \sum_{i=1}^n (b x_i + a - y_i)^2.$$

Todos os x_i e y_i na função são conhecidos (são os dados).

O que fica para definir são b e a .

b e a são comuns a todos os termos; é necessário achar um valor único que torne a soma o menor número possível.

O problema é, para cada ponto a função tem um termo.

Por exemplo, para $D = \{\{4, 5\}, \{2, 7\}, \{9, 1\}, \{12, 6\}, \{5, 3\}\}$.

$$(4b + a - 5)^2 + (2b + a - 7)^2 + (9b + a - 1)^2 + (12b + a - 6)^2 + (5b + a - 3)^2 =$$

```
In[*]:= {Expand[(4 b + a - 5)^2], Expand[(2 b + a - 7)^2],
Expand[(9 b + a - 1)^2], Expand[(12 b + a - 6)^2], Expand[(5 b + a - 3)^2]}
```

```
Out[*]:= {25 - 10 a + a^2 - 40 b + 8 a b + 16 b^2, 49 - 14 a + a^2 - 28 b + 4 a b + 4 b^2,
1 - 2 a + a^2 - 18 b + 18 a b + 81 b^2, 36 - 12 a + a^2 - 144 b + 24 a b + 144 b^2, 9 - 6 a + a^2 - 30 b + 10 a b + 25 b^2}
```

Somando os termos...

```
In[*]:= Expand[(4 b + a - 5)^2] + Expand[(2 b + a - 7)^2] +
Expand[(9 b + a - 1)^2] + Expand[(12 b + a - 6)^2] + Expand[(5 b + a - 3)^2]
```

```
Out[*]:= 120 - 44 a + 5 a^2 - 260 b + 64 a b + 270 b^2
```

Agora temos termos a , b , a^2 , b^2 e $a b$ (e um termo livre).

Para cada um destes termos, temos um coeficiente.

```
In[*]:= FullSimplify[120 - 44 a + 5 a^2 - 260 b + 64 a b + 270 b^2]
```

```
Out[*]:= 120 + a (-44 + 5 a) - 260 b + 64 a b + 270 b^2
```

Não dá para agrupar só em a e b porque temos um termo $a b$.

Se não tivéssemos este termo,

```
In[*]:= FullSimplify[120 - 44 a + 5 a^2 - 260 b + 270 b^2]
```

```
Out[*]:= 120 + a (-44 + 5 a) + 10 b (-26 + 27 b)
```

Mesmo assim temos as potências.

Mas a função geral para minimizar para a e b é

$$120 - 44a + 5a^2 - 260b + 64ab + 270b^2.$$

Se não fosse a diferença ao quadrado, apenas simples,

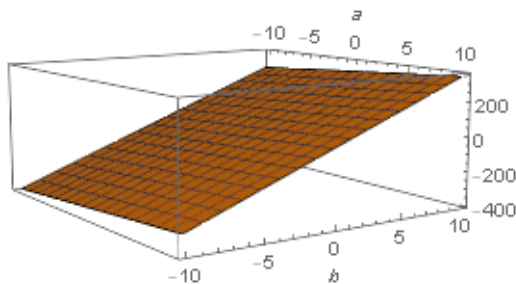
```
In[*]:= 4 b + a - 5 + 2 b + a - 7 + 9 b + a - 1 + 12 b + a - 6 + 5 b + a - 3
```

```
Out[*]:= -22 + 5 a + 32 b
```

Ou seja, é o quadrado da diferença que introduz essa “multiplicidade de variáveis”.

Começar com a simples, $-22 + 5a + 32b$, para minimizar.

```
In[*]:= Plot3D[32 b + 5 a - 22, {a, -10, 10}, {b, -10, 10}, AxesLabel -> Automatic]
```



Derivadas parciais.

Mantendo b constante, $\frac{\partial y}{\partial a} = 5$ (pois se b é **uma** constante, $32b$ é derivado como 0);

mantendo a constante, $\frac{\partial y}{\partial b} = 32$ (pois se a é **uma** constante, $5a$ é derivado como 0).

```
In[*]:= {D[32 b + 5 a - 22, a], D[32 b + 5 a - 22, b]}
```

```
Out[*]:= {5, 32}
```

Então as diferenças simples têm parciais constantes.

Mas os extremos são igualar a zero.

Mas não dá para igualar estas derivadas a zero; elas são os próprios extremos.

Pausando para uma função qualquer arbitrária.

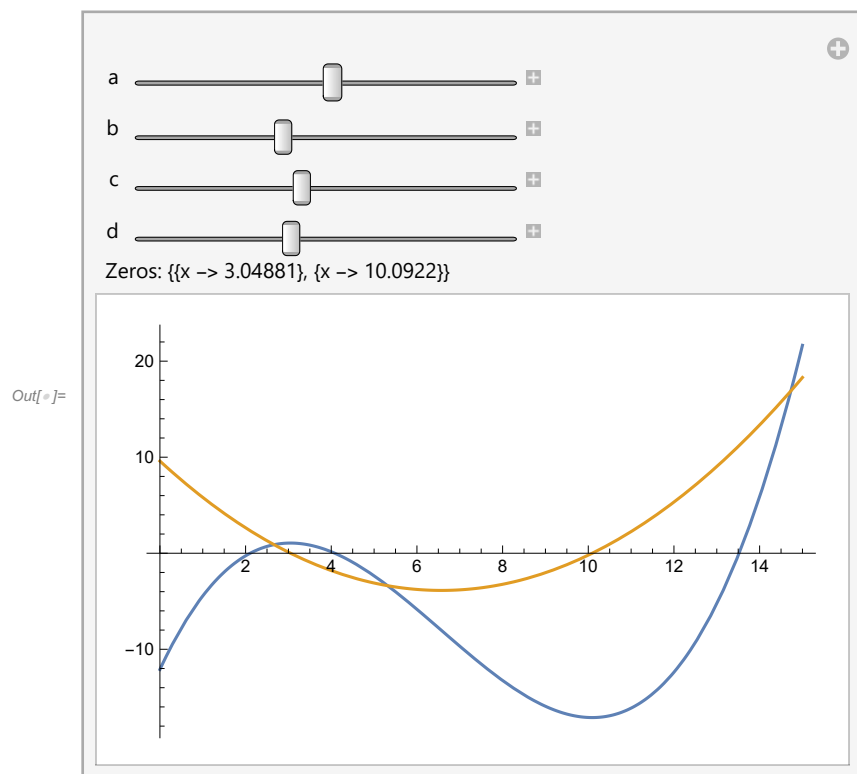
```
In[*]:= D[0.1 x^3 - 2 x^2 + 10.3 x - 8.1, x]
```

```
Out[*]:= 10.3 - 4 x + 0.3 x^2
```

```

In[ ]:= Module[{f, df}, Manipulate[f = a x^3 + b x^2 + c x + d;
  df = D[f, x];
  Plot[{f, df}, {x, 0, 15} (*, PlotRange -> {{0, 15}, {-10, 20}} *)],
  {{a, 0.1}, 0, 0.2, 0.001}, {{b, -2}, -2.2, -1.8, .01}, {{c, 10.3}, 5.3, 15.3, .1},
  {{d, -8.1}, -28.1, 12.1, .1}, Dynamic["Zeros: " <> ToString[Solve[df == 0, x]]]]

```



Mas estes não são extremos, são pontos de inflexão...

Nestas raízes, y é...

```

In[ ]:= Clear[f1, f1ext]
f1=Function[x, 0.1 x^3 - 2 x^2 + 10.3 x - 8.1];
f1ext=Solve[D[f1[x]] == 0, x]

```

```
Out[ ]:= {{x -> 3.48687}, {x -> 9.84646}}
```

```
In[ ]:= {f1[x /. First[f1ext]], f1[x /. Last[f1ext]]}
```

```
Out[ ]:= {7.73766, -5.12285}
```

Sem usar os dados, as derivadas parciais da soma da minimização dos quadrados da diferença

$\partial_a \sum_{i=1}^n (b x_i + a - y_i)^2$ e $\partial_b \sum_{i=1}^n (b x_i + a - y_i)^2$ seriam

$$\text{In[]:= } \sum_{i=1}^n (b x_i + a - y_i)$$

$$\text{Out[]:= } \sum_{i=1}^n (a + b x_i - y_i)$$

$$\text{In[]:= } \left\{ \partial_a \sum_{i=1}^n (b x_i + a - y_i)^2, \partial_b \sum_{i=1}^n (b x_i + a - y_i)^2 \right\}$$

$$\text{Out[]:= } \left\{ \sum_{i=1}^n (2 a + 2 b x_i - 2 y_i), \sum_{i=1}^n (2 a x_i + 2 b x_i^2 - 2 x_i y_i) \right\}$$

Sem os quadrados...

$$\text{In}[*]:= \left\{ \partial_a \sum_{i=1}^n b x_i + a - y_i, \partial_b \sum_{i=1}^n b x_i + a - y_i \right\}$$

$$\text{Out}[*]:= \left\{ a - y_i, a - y_i + \sum_{i=1}^n x_i \right\}$$

Igualar a zero...

$$\text{In}[*]:= \text{Solve} \left[\sum_{i=1}^n (2 a + 2 b x_i - 2 y_i) == 0, b \right]$$

... Solve: This system cannot be solved with the methods available to Solve.

$$\text{Out}[*]:= \text{Solve} \left[\sum_{i=1}^n (2 a + 2 b x_i - 2 y_i) == 0, b \right]$$

Usando os dados D na diferença sem quadrados, as diferenças simples têm parciais constantes. Nestas derivadas (números), y é

```
In[*]:= Clear[f2, f2ext]
f2=Function[{a,b}, 32b+5a-22];
f2ext=<|"a"→∂af2[a,b], "b"→∂bf2[a,b] |>
```

```
Out[*]:= <| a → 5, b → 32 |>
```

A função é bidimensional e tem os extremos para cada variável/dimensão. Logo devemos plugar estes extremos na função bidimensional?

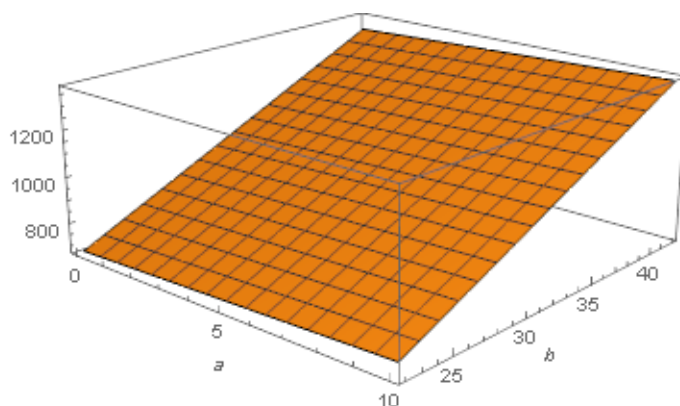
```
In[*]:= f2[f2ext["a"], f2ext["b"]]
```

```
Out[*]:= 1027
```

```
In[*]:= Dt[f2[a, b]]
```

```
Out[*]:= 5 Dt[a] + 32 Dt[b]
```

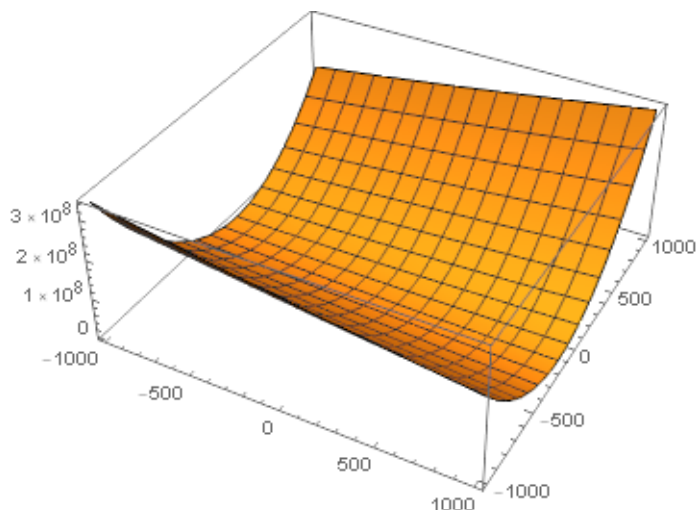
```
In[*]:= Plot3D[32 b + 5 a - 22, {a, 0, 10}, {b, 22, 42}, AxesLabel → Automatic(*, Epilog → Line[{5, 32}]*)]
```



Talvez por isso a diferença é ao quadrado... Função linear não tem extremo.

Visualizar a diferença ao quadrado.

```
In[1]:= Clear[f3]
f3=Function[{a,b}, 120-44 a+5 a^2-260 b+64 a b+270 b^2];
Plot3D[f3[a,b], {a, -1000, 1000}, {b, -1000, 1000}]
```



```
In[*]:= {D[f3[a, b], a], D[f3[a, b], b]}
```

```
Out[*]:= {-44 + 10 a + 64 b, -260 + 64 a + 540 b}
```

```
In[*]:= {Solve[D[f3[a, b], a] == 0, a], Solve[D[f3[a, b], b] == 0, b]}
```

```
Out[*]:= {{a -> -2/5 (-11 + 16 b)}, {b -> 1/135 (65 - 16 a)}}
```

Invertido? (Não sei o significado.)

```
In[*]:= {Solve[D[f3[a, b], b] == 0, b], Solve[D[f3[a, b], a] == 0, a]}
```

```
Out[*]:= {{b -> 1/32 (22 - 5 a)}, {a -> 5/16 (-13 + 27 b)}}
```

As derivadas parciais das bidimensionais ainda são bidimensionais, para encontrar os extremos (números), encontrar a solução comum das equações parciais, o que monta a matriz.

```
In[*]:= Solve[(D[f3[a, b], a] == 0) && (D[f3[a, b], b] == 0), a]
```

```
Out[*]:= {}
```

```
In[*]:= Solve[(-44 + 10 a + 64 b == 0) && (-260 + 64 a + 540 b == 0), a]
```

```
Out[*]:= {}
```

```
In[*]:= Solve[(D[f3[a, b], b] == 0) && (D[f3[a, b], a] == 0), b]
```

```
Out[*]:= {}
```

```
In[*]:= Solve[(-44 + 10 a + 64 b == 0) && (-260 + 64 a + 540 b == 0), b]
```

```
Out[*]:= {}
```

Talvez não tenha solução por causa destes dados.

O 2 antes das derivadas parciais é por causa da diferenciação do quadrado¹. E o X_1 multiplicado extra na b ?

```
In[*]:= {D[b x1 + a - y1]^2, D[a (b x1 + a - y1)^2]}
```

```
Out[*]:= {2 x1 (a + b x1 - y1), 2 (a + b x1 - y1)}
```

Chain rule. A somatória de uma soma pode ser a soma das somatórias. 2

$$\begin{aligned}\partial_a \sum_{i=1}^n (b x_i + a - y_i)^2 &= \\ \sum_{i=1}^n (2a + 2b x_i - 2y_i) &= \\ 2na + 2 \sum_{i=1}^n b x_i - 2 \sum_{i=1}^n y_i &= \\ 2an + 2b \sum_{i=1}^n x_i - 2 \sum_{i=1}^n y_i\end{aligned}$$

e

$$\begin{aligned}\partial_b \sum_{i=1}^n (b x_i + a - y_i)^2 &= \\ \sum_{i=1}^n (2a x_i + 2b x_i^2 - 2x_i y_i) &= \\ 2 \sum_{i=1}^n a x_i + 2 \sum_{i=1}^n b x_i^2 - 2 \sum_{i=1}^n y_i x_i &= \\ 2a \sum_{i=1}^n x_i + 2b \sum_{i=1}^n x_i^2 - 2 \sum_{i=1}^n y_i x_i.\end{aligned}$$

Em ∂_a , por algum motivo, $\sum_{i=1}^n a$ vira na .

Mas este não é o principal; mesmo na forma básica, as parciais já contam com uma separação das variáveis.

Igualando a zero,

$$\begin{aligned}2an + 2b \sum_{i=1}^n x_i - 2 \sum_{i=1}^n y_i &= 0 \Rightarrow \\ 2an + 2b \sum_{i=1}^n x_i &= 2 \sum_{i=1}^n y_i\end{aligned}$$

e

$$\begin{aligned}2a \sum_{i=1}^n x_i + 2b \sum_{i=1}^n x_i^2 - 2 \sum_{i=1}^n y_i x_i &= 0 \Rightarrow \\ 2a \sum_{i=1}^n x_i + 2b \sum_{i=1}^n x_i^2 &= 2 \sum_{i=1}^n y_i x_i.\end{aligned}$$

Dividindo ambos por 2,

$$\begin{aligned}an + b \sum_{i=1}^n x_i &= \sum_{i=1}^n y_i \text{ e} \\ a \sum_{i=1}^n x_i + b \sum_{i=1}^n x_i^2 &= \sum_{i=1}^n y_i x_i.\end{aligned}$$

Este sistema é na forma

$$an + br = s \text{ e}$$

$$ar + bt = u, \text{ onde}$$

$$r = \sum_{i=1}^n x_i,$$

$$s = \sum_{i=1}^n y_i,$$

$$t = \sum_{i=1}^n x_i^2,$$

$$u = \sum_{i=1}^n y_i x_i.$$

Somando,

$$an + br + ar + bt = s + u$$

Para isolar b ,

$$b(r + t) = s + u - a(n - r) \Rightarrow$$

$$b = \frac{s + u - a(n - r)}{r + t}$$

e a ,

$$a(r+t) = s + u - b(r+t) \Rightarrow$$

$$a = \frac{s+u-b(r+t)}{r+t}.$$

Para resolver o sistema, antes de somar,

$$r(an + br) = rs_e$$

$$-n(ar + bt) = -nu,$$

$$arn + br^2 - anr - bnt = rs - nu \Rightarrow$$

$$b(r^2 - nt) = rs - nu \Rightarrow$$

$$b = \frac{rs - nu}{r^2 - nt}.$$

Reescrevendo a primeira equação por a ,

$$an + br = s \Rightarrow$$

$$an = s - br \Rightarrow$$

$$a = \frac{s - br}{n}.$$

O paper pára aqui (sem substituir $\frac{rs - nu}{r^2 - nt}$ como b).

As equações normais são as equações das variáveis em função de X_i e $y_i - \hat{y}_i$, ou seja, a diferença entre o modelo e os dados.

(As variáveis compõem os coeficientes do modelo.)

Estas equações são o mínimo de cada variável em função da função diferença dados/modelo.

(Ou seja, igualar a derivada parcial de cada variável a zero.)

A função da diferença dados/modelo “como um todo” só tem mínimo se *todas* as variáveis ou derivadas parciais têm mínimo *simultaneamente*. Por isso o mínimo de cada variável é inputado em um sistema que deve ter resolução simultânea para todas as variáveis.

A função da diferença dados/modelo é a soma ou somatória de n instâncias da função diferença dado/-modelo com X_i e y_i substituídos na função, vindos dos dados, e \hat{y}_i substituído na função vindo do modelo, com n a quantidade de pontos nos dados.

(Ou seja, a soma de todas as funções diferença modelo/dado determina os *coeficientes desta função*, que são os valores que compõem o sistema para solução.)

Na prática, como as variáveis são as incógnitas, são as somas dos X e $y - \hat{y}$ dos dados e modelo que definem os coeficientes do sistema.

Ao resolver o sistema, temos os valores do mínimo de cada variável para a função modelo + diferença dados/modelo, o que são os coeficientes que compõem o modelo que minimiza estes números.

O seguinte dado com fit cosseno manual em β e γ resulta nos seguintes coeficientes:

$$\beta = 0.4$$

$$\gamma = 0.7$$

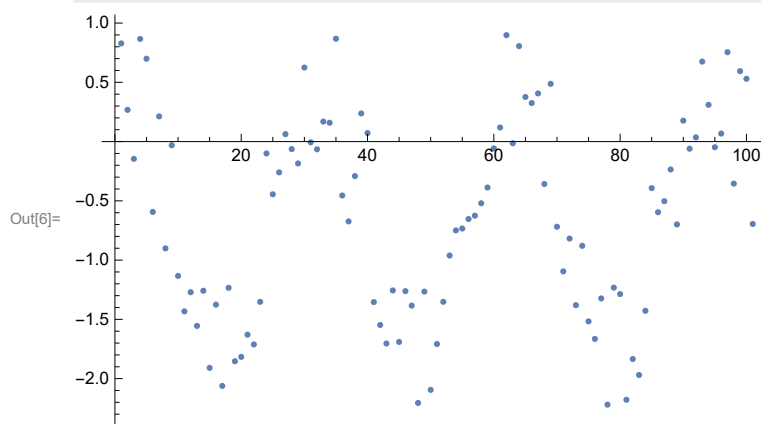
$$\tau = 35.4$$

$$M = -0.6.$$

Com erro ao quadrado 39.0437.

```
In[ ]:= (*isso deve sempre ser verdadeiro*)
Clear[M, τ, β, γ]
```

```
In[4]:= (*Modeling.nb points2b2*)
Clear[points1, vpar1]
points1={0.8284486071053614`, 0.2673766376785629`, -0.14587103848031902`, 0.8659704680415028`, 0.69910541126
ListPlot[points1, ImageSize→Medium]
vpar1=<|"β"→0.4, "γ"→0.7, "τ"→35.4, "M"→-0.6|>;
```



A função do modelo é

$$\hat{Y}(t) = M + A \cos\left(\frac{2\pi t}{\tau} + \phi\right), \text{ com a identidade trigonométrica,}$$

$$\hat{Y}(t) = M + A \cos(\phi) \cos\left(\frac{2\pi t}{\tau}\right) - A \sin(\phi) \sin\left(\frac{2\pi t}{\tau}\right), \text{ com os parâmetros,}$$

$$\hat{Y}(t) = M + \beta \cos\left(\frac{2\pi t}{\tau}\right) - \gamma \sin\left(\frac{2\pi t}{\tau}\right).$$

Tomando em alguns pontos

```
In[ ]:= Clear[testt1, fmodelparam1]
testt1={11, 30, 50, 65};
(*fmodelparam1 dá o modelo em um ponto*)
(*t na função modelo é baseado em 0, e não 1*)
fmodelparam1=Function[{t, M, τ, β, γ}, M+β Cos[2 π t / τ]-γ Sin[2 π t / τ]]
```

```
Out[ ]:= Function[{t, M, τ, β, γ}, M+β Cos[2 π t / τ]-γ Sin[2 π t / τ]]
```

```
In[ ]:= Table[fmodelparam1[t, vpar1["M"], vpar1["τ"], vpar1["β"], vpar1["γ"]], {t, testt1}]
```

```
Out[ ]:= {-1.39861, 0.202727, -1.30698, 0.206027}
```

```
In[ ]:= fmodelparam1[10, vpar1["M"], vpar1["τ"], β, γ]
```

```
Out[ ]:= -0.6 - 0.2027 β - 0.979241 γ
```

Simulando dados nestes pontos


```
In[ ]:= Clear[testdata1]
testdata1 = {{11, -1}, {30, 0.5}, {50, -1.4}, {65, 0.1}};
```

A função diferença dados/modelo é

$$\Delta Y(t) = \sum_{i=1}^n y_i - \left[M + \beta \cos\left(\frac{2\pi t_i}{\tau}\right) - \gamma \sin\left(\frac{2\pi t_i}{\tau}\right) \right]$$

```
Clear[fmodeldif1part1, fmodeldif1part2, fmodeldif1indiv, fmodeldifsq1indiv,
fmodeldif1, fmodeldifsq1]
(*testando a somatória*)
fmodeldif1part1 = Function[{t, data}, Sum[data[[i]][[2]], {i, 1, Length[data]}];
fmodeldif1part2 = Function[{t, data},
Sum[fmodelparam1[t[[i]], vpar1["M"], vpar1["tau"], beta, gamma], {i, 1, Length[data]}];
(*teste de diferença em um ponto*)
(*fmodeldif1indiv dá a diferença entre um dado e o modelo em um ponto*)
fmodeldif1indiv = Function[{M, tau, beta, gamma, t, y}, y - fmodelparam1[t, M, tau, beta, gamma]];
fmodeldifsq1indiv = Function[{M, tau, beta, gamma, t, y}, (y - fmodelparam1[t, M, tau, beta, gamma])^2];
(*a(s) função(ões)*)
(*estas funções, sem os parâmetros definidos, dão as funções a minimizar
(igualar as parciais a zero) -- ainda separadas por termo -- usar
Simplify[] --, e com os parâmetros, as somas dos resíduos.*)
(*t é uma lista de posições (x também chamado t), com início em 0, que
também deve constar em data, que são pontos (x,y)*)
fmodeldif1 = Function[{M, tau, beta, gamma, t, data},
Sum[(data[[i]][[2]] - fmodelparam1[t[[i]], M, tau, beta, gamma]), {i, 1, Length[data]}];
fmodeldifsq1 = Function[{M, tau, beta, gamma, t, data},
Sum[(data[[i]][[2]] - fmodelparam1[t[[i]], M, tau, beta, gamma])^2, {i, 1, Length[data]}];
```

Testes da somatória...

```
In[ ]:= {fmodeldif1part1[testtt1, testdata1], fmodeldif1part2[testtt1, testdata1]}
```

```
Out[ ]:= {-1.8, -2.4 - 0.134741 beta + 0.224373 gamma}
```

Testando as diferenças para os pontos...

São tantas diferenças quanto pontos.

Aqui estou passando índice baseado em 1 para testtt1 (ao invés de baseado 0) porque ele não é um ts, é uma lista contendo as posições. (?)

```
In[ ]:= Print[Column[Table[fmodeldif1indiv[vpar1["M"], vpar1["tau"], beta,
gamma, testtt1[[i + 1]], testdata1[[i + 1]][[2]]], {i, 0, Length[testtt1] - 1}]]]
-0.4 + 0.372411 beta + 0.928068 gamma
1.1 - 0.574787 beta - 0.818303 gamma
-0.8 + 0.852408 beta + 0.522877 gamma
0.7 - 0.515292 beta - 0.857015 gamma
```

Soma das diferenças...

```
In[ ]:= -0.4 + 0.372411 beta + 0.928068 gamma + 1.1 - 0.574787 beta -
0.818303 gamma + -0.8 + 0.852408 beta + 0.522877 gamma + 0.7 - 0.515292 beta - 0.857015 gamma
Out[ ]:= 0.6 + 0.13474 beta - 0.224373 gamma
```

Soma das diferenças via somatória.

```
In[ ]:= Clear[residual1]
residual1 = fmodeldif1[vpar1["M"], vpar1["τ"], β, γ, testt1, testdata1];
Simplify[residual1]

Out[ ]:= 0.6 + 0.134741 β - 0.224373 γ
```

Bateu. Minimizar em β e γ .

```
In[ ]:= {Dβ residual1, Dγ residual1}

Out[ ]:= {0.134741, -0.224373}
```

Estou tendo as derivadas constantes porque estou fazendo erro linear e não quadrático...

```
In[ ]:= Clear[residualsq1];
residualsq1=fmodeldifsq1[vpar1["M"],vpar1["τ"],β,γ,testt1,testdata1];
Print[residualsq1];
Simplify[residualsq1]

(0.7 - 0.515292 β - 0.857015 γ)2 + (1.1 - 0.574787 β - 0.818303 γ)2 +
(-0.8 + 0.852408 β + 0.522877 γ)2 + (-0.4 + 0.372411 β + 0.928068 γ)2

Out[ ]:= 2.5 + 1.4612 β2 - 4.57914 γ + 2.5388 γ2 + β (-3.64772 + 3.40658 γ)
```

A somatória veio sem somar.

Minimizar em β e γ .

```
In[ ]:= Print[Column[{Dβ residualsq1, "\n", Dγ residualsq1}]]

-1.03058 (0.7 - 0.515292 β - 0.857015 γ) - 1.14957 (1.1 - 0.574787 β - 0.818303 γ) +
1.70482 (-0.8 + 0.852408 β + 0.522877 γ) + 0.744823 (-0.4 + 0.372411 β + 0.928068 γ)

-1.71403 (0.7 - 0.515292 β - 0.857015 γ) - 1.63661 (1.1 - 0.574787 β - 0.818303 γ) +
1.04575 (-0.8 + 0.852408 β + 0.522877 γ) + 1.85614 (-0.4 + 0.372411 β + 0.928068 γ)
```

Vem sem somar, somando...

```
In[ ]:= Clear[partials1]
partials1={Simplify[Dβ residualsq1],Simplify[Dγ residualsq1]};
Print[Column[{partials1[[1]],"\n",partials1[[2]]}]]

-3.64772 + 2.92239 β + 3.40658 γ

-4.57914 + 3.40658 β + 5.07761 γ
```

Estas são as duas parciais, uma em cada variável.

As duas têm que ser igualadas a zero simultaneamente.

```
In[ ]:= Clear[minpars1]
minpars1=Solve[partials1[[1]]==0&&partials1[[2]]==0]

Out[ ]:= {{β → 0.903683, γ → 0.295548}}
```

Estes parâmetros substituem o β e γ definidos visualmente, mas mantendo o M e τ .

O erro ao quadrado para os parâmetros definidos visualmente é

```
In[ ]:= Clear[residual1v]
residual1v=fmodeldifsq1[vpar1["M"],vpar1["τ"],vpar1["β"],vpar1["γ"],testt1,testdata1];
residual1v

Out[ ]:= 0.267158
```

```
In[*]:= Clear[difindivs1v]
difindivs1v = Table[fmodeldifsq1indiv[vpar1["M"], vpar1["τ"],
  vpar1["β"], vpar1["γ"], testt1[[i]], testdata1[[i]][[2]]], {i, Length[testt1]}]
Out[*]:= {0.158892, 0.0883713, 0.00865326, 0.0112417}
```

```
In[*]:= Total[difindivs1v]
Out[*]:= 0.267158
```

Batendo as diferenças manualmente...

```
In[*]:= {(-1 + 1.3986119758301152)^2, (0.5 - 0.20272690722148112)^2,
  (-1.4 + 1.3069770906904363)^2, (0.1 - 0.2060270811124157)^2}
Out[*]:= {0.158892, 0.0883713, 0.00865326, 0.0112417}
```

Estar diferente da diferença no modelo original (39.0437) é por causa dos dados.

As diferenças com os parâmetros minimizados...

```
In[*]:= difindivs1 =
  Table[fmodeldifsq1indiv[vpar1["M"], vpar1["τ"], β, γ, testt1[[i]], testdata1[[i]][[2]]] /.
    First[minpars1], {i, Length[testt1]}]
Out[*]:= {0.0444494, 0.114736, 0.0155855, 0.000359067}
```

```
In[*]:= Total[difindivs1]
Out[*]:= 0.17513
```

Diminuiu!

```
In[*]:= Clear[residual1]
residual1 = fmodeldifsq1[vpar1["M"], vpar1["τ"], β, γ, testt1, testdata1] /. First[minpars1];
residual1
Out[*]:= 0.17513
```

Agora, usando os dados.

Minimizar β e γ . Pegar a somatória do resíduo sem estas variáveis.

```
In[*]:= fmodelparam1
Out[*]:= Function[{t, M, τ, β, γ}, M + β Cos[ $\frac{2\pi t}{\tau}$ ] - γ Sin[ $\frac{2\pi t}{\tau}$ ]]

In[*]:= fmodelparam1[0, vpar1["M"], vpar1["τ"], vpar1["β"], vpar1["γ"]]
Out[*]:= -0.2
```

```
In[*]:= fmodeldif1indiv
Out[*]:= Function[{M, τ, β, γ, t, y}, y - fmodelparam1[t, M, τ, β, γ]]
```

```
In[*]:= fmodeldif1
Out[*]:= Function[{M, τ, β, γ, t, data},  $\sum_{i=1}^{\text{Length[data]}} (\text{data}[[i]][[2]] - \text{fmodelparam1}[t[[i]], M, \tau, \beta, \gamma])$ ]
```

Nas chamadas ao fmodelparam1, passar t baseado em 0.

Como uso o ts para passar, montar o ts a partir de 0.

```
In[*]:= Module[{dta, dtapoints, mdl, mdlpoints, ts, difsq, rss,
  minrssf, minrssp, minrssl, minrss, Ahat, K, φhat},
  (*definir o dado, criar o ts, "discretizar" ou samplear o modelo*)
  dta = points1;
  (*Print[dta];*)
```

```

ts = Range[0, Length[dta] - 1]; (*já gerar o ts baseado no dta*)
(*gerar o dtapoints baseado no ts (base 0) e dta (base 1)*)
dtapoints = Table[{i, dta[i + 1]}, {i, ts}];
(*Print[dtapoints];*)
(*discretize*)
mdl = Table[
  fmodelparam1[t, vpar1["M"], vpar1["τ"], vpar1["β"], vpar1["γ"]], {t, 0, Length[dta] - 1}];
(*checado*)
(*Print[mdl];*)
Print[
  ListLinePlot[{dta, mdl}, ImageSize → Medium, PlotRange → {{0, Length[dta] - 1}, {-3, 3}}]];
(*Print[{Length[dta], Length[dtapoints], Length[mdl], Length[ts]}];*)

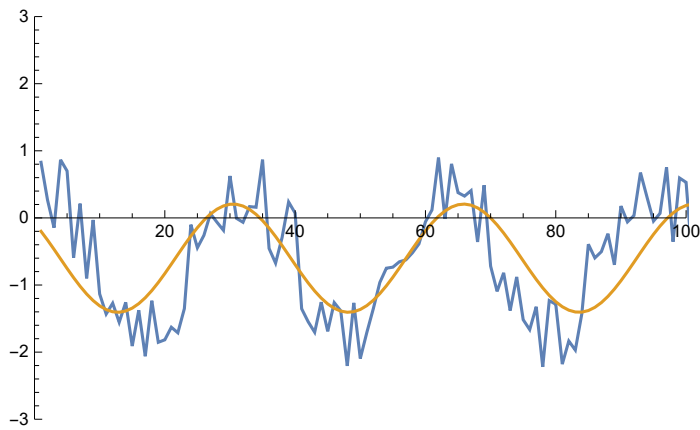
(*calcular o resíduo visual*)
rss = fmodeldifsq1[vpar1["M"], vpar1["τ"], vpar1["β"], vpar1["γ"], ts, dtapoints];
Print["visual rss: ", rss];
(*testes de debug de rss*)
(*Print[fmodelparam1[ts[[1]], vpar1["M"], vpar1["τ"], vpar1["β"], vpar1["γ"]];*)
(*Print["first data is ", First[dta]];
Print["first model is ", First[mdl]];
Print["first res should be ", ToString[0.828449+0.2]];
Print["first res is ",
  fmodeldif1indiv[vpar1["M"], vpar1["τ"], vpar1["β"], vpar1["γ"], First[ts], First[dta]]];
Print["first res sq² is ", fmodeldifsq1indiv[vpar1["M"],
  vpar1["τ"], vpar1["β"], vpar1["γ"], First[ts], First[dta]]];*)

(*calcular o resíduo mínimo*)
(*paper: τ não tem estimativa.*)
minrssf = fmodeldifsq1[M, vpar1["τ"], β, γ, ts, dtapoints];
Print["min rss fun: ", Simplify[minrssf]];
minrsspar = <|
  "M" → Simplify[∂Mminrssf],
  "β" → Simplify[∂βminrssf],
  "γ" → Simplify[∂γminrssf]
|>;
Print["min rss par: ", minrsspar];
minrssl = Solve[
  minrsspar["M"] == 0 &&
  minrsspar["β"] == 0 &&
  minrsspar["γ"] == 0
];
Print["min rss solution: ", minrssl];
minrss = minrssf /. minrssl;
Print["min rss: ", minrss];

(*plotar o minimizado*)
(*TODO: porquê necessário First? abaixo*)
minmdlf = First[fmodelparam1[t, M, vpar1["τ"], β, γ] /. minrssl];
Print["min model fun: ", minmdlf];
(*Plot[minmdlf, {t, 0, 100}];*)
minmdl = Table[minmdlf, {t, 0, Length[dta] - 1}];
Print[
  ListLinePlot[{mdl, minmdl}, ImageSize → Medium, PlotRange → {{0, Length[dta] - 1}, {-3, 3}}]];

(*estimar amplitude e acrofase*)
Ahat =  $\sqrt{\beta^2 + \gamma^2}$  /. minrssl;
K = 1;
φhat = ArcTan[ $\frac{-\gamma}{\beta}$ ] + K π /. minrssl;
Print["Â: ", Ahat];
Print["φ̂: ", φhat];

```



visual rss: 39.0437

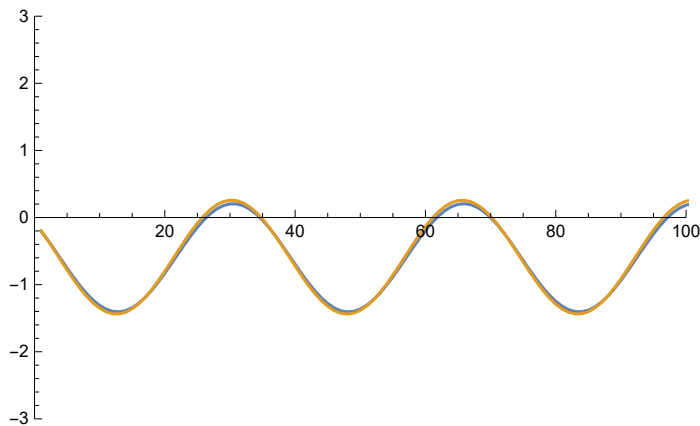
min rss fun: 101. $(1.12753 + M^2 + 0.489868 \beta^2 +$
 $M (1.25139 - 0.0848016 \beta - 0.0520183 \gamma) + \beta (-0.405116 - 0.0398594 \gamma) - 0.782083 \gamma + 0.510132 \gamma^2)$

min rss par: $\langle M \rightarrow 202. (0.625696 + M - 0.0424008 \beta - 0.0260091 \gamma),$
 $\beta \rightarrow -40.9167 - 8.56497 M + 98.9532 \beta - 4.0258 \gamma, \gamma \rightarrow -78.9904 - 5.25385 M - 4.0258 \beta + 103.047 \gamma \rangle$

min rss solution: $\{ \{ M \rightarrow -0.589475, \beta \rightarrow 0.393061, \gamma \rightarrow 0.751851 \} \}$

min rss: {38.8924}

min model fun: $-0.589475 + 0.393061 \text{Cos}[0.177491 t] - 0.751851 \text{Sin}[0.177491 t]$



\hat{A} : {0.848397}

$\hat{\phi}$: {2.05251}

TODO: exibir (e resolver) a NE (equações normais) em forma matricial.
 Valores M, τ, β, γ estão vazando para o global e contaminando as funções.

Testes de **ts**.

Aparentemente, mudar a razão de **ts** só vai escalonar o eixo X .

O problema é usar o valor de **ts** para acessar os índices dos dados.

Os dados têm de ser acessados pelos índices de **ts**, não por seus valores.

Mas neste caso, **ts** não vai servir para mais absolutamente nada.

Ou, criar os pontos (indexar) dos dados por **ts**.

In[*]:= **fmodelparam1**

Out[*]:= **Function** $\left[\{t, M, \tau, \beta, \gamma\}, M + \beta \text{Cos}\left[\frac{2 \pi t}{\tau}\right] - \gamma \text{Sin}\left[\frac{2 \pi t}{\tau}\right] \right]$

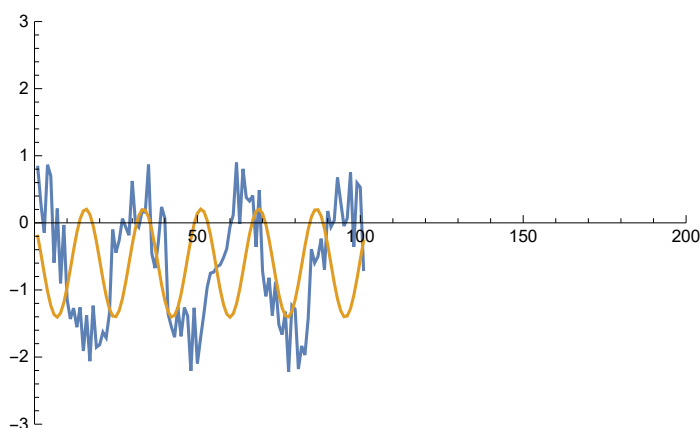
```

In[ ]:= Module[{dta, ts, dtapoints},
  dta = points1;
  ts = Range[0, (Length[dta] - 1) * 2, 2];
  Print[Length[ts]];
  Print[ts];
  dtapoints = Table[{i - 1, dta[[i]]}, {i, Range[1, Length[ts]]}];
  Print[dtapoints];
  mdl = Table[fmodelparam1[t, vpar1["M"], vpar1[" $\tau$ "], vpar1[" $\beta$ "], vpar1[" $\gamma$ "]], {t, ts}];
  Print[mdl];
  Print[ListLinePlot[{dta, mdl}, ImageSize -> Medium, PlotRange -> {{0, 200}, {-3, 3}}]];
];
101
{0, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36, 38, 40, 42, 44,
46, 48, 50, 52, 54, 56, 58, 60, 62, 64, 66, 68, 70, 72, 74, 76, 78, 80, 82, 84, 86, 88, 90,
92, 94, 96, 98, 100, 102, 104, 106, 108, 110, 112, 114, 116, 118, 120, 122, 124, 126, 128,
130, 132, 134, 136, 138, 140, 142, 144, 146, 148, 150, 152, 154, 156, 158, 160, 162, 164,
166, 168, 170, 172, 174, 176, 178, 180, 182, 184, 186, 188, 190, 192, 194, 196, 198, 200}

{{0, 0.828449}, {1, 0.267377}, {2, -0.145871}, {3, 0.86597}, {4, 0.699105}, {5, -0.593286},
{6, 0.21297}, {7, -0.901046}, {8, -0.0308795}, {9, -1.13305}, {10, -1.43241}, {11, -1.27098},
{12, -1.55522}, {13, -1.25866}, {14, -1.90932}, {15, -1.3759}, {16, -2.06161}, {17, -1.23359},
{18, -1.85334}, {19, -1.81647}, {20, -1.62901}, {21, -1.71166}, {22, -1.35207}, {23, -0.10037},
{24, -0.444725}, {25, -0.259657}, {26, 0.0628543}, {27, -0.0634696}, {28, -0.184484},
{29, 0.624653}, {30, -0.00617488}, {31, -0.0650214}, {32, 0.168654}, {33, 0.158918}, {34, 0.867811},
{35, -0.454881}, {36, -0.673642}, {37, -0.291191}, {38, 0.237324}, {39, 0.0717739}, {40, -1.35391},
{41, -1.54713}, {42, -1.70356}, {43, -1.25588}, {44, -1.69065}, {45, -1.26194}, {46, -1.38428},
{47, -2.20528}, {48, -1.26573}, {49, -2.0948}, {50, -1.70808}, {51, -1.35185}, {52, -0.961597},
{53, -0.749494}, {54, -0.733854}, {55, -0.653462}, {56, -0.625239}, {57, -0.520483}, {58, -0.387476},
{59, -0.0575814}, {60, 0.118542}, {61, 0.898165}, {62, -0.0139777}, {63, 0.805251}, {64, 0.37662},
{65, 0.325103}, {66, 0.406461}, {67, -0.358079}, {68, 0.487177}, {69, -0.718652}, {70, -1.09497},
{71, -0.818308}, {72, -1.38016}, {73, -0.879561}, {74, -1.51713}, {75, -1.66477}, {76, -1.32329},
{77, -2.21958}, {78, -1.2324}, {79, -1.2868}, {80, -2.17856}, {81, -1.83439}, {82, -1.96933},
{83, -1.4279}, {84, -0.392676}, {85, -0.596141}, {86, -0.503018}, {87, -0.235215}, {88, -0.698701},
{89, 0.176379}, {90, -0.0599401}, {91, 0.0354239}, {92, 0.674987}, {93, 0.310189}, {94, -0.047734},
{95, 0.0676676}, {96, 0.754158}, {97, -0.354738}, {98, 0.594251}, {99, 0.529957}, {100, -0.695208}}

{-0.2, -0.468241, -0.752911, -1.01851, -1.23193, -1.36655, -1.40558, -1.34416, -1.18995, -0.962178,
-0.689241, -0.405176, -0.145404, 0.0576812, 0.178758, 0.202727, 0.126601, -0.040129, -0.276672,
-0.553531, -0.836186, -1.08939, -1.28157, -1.38876, -1.3976, -1.30698, -1.1282, -0.883563, -0.603565,
-0.323123, -0.0772054, 0.103522, 0.196524, 0.190204, 0.0853496, -0.104964, -0.357006, -0.639349,
-0.916784, -1.15472, -1.32348, -1.40203, -1.38057, -1.26178, -1.06047, -0.801735, -0.517849,
-0.244207, -0.0149298, 0.141392, 0.205266, 0.168728, 0.0363334, -0.175408, -0.440095, -0.72472,
-0.993794, -1.21376, -1.3572, -1.40622, -1.3547, -1.20908, -0.987513, -0.717622, -0.433064,
-0.169322, 0.0407168, 0.170862, 0.204884, 0.138542, -0.0198927, -0.250663, -0.524995, -0.808679,
-1.06634, -1.26585, -1.38234, -1.40127, -1.32029, -1.14949, -0.91017, -0.632176, -0.35017,
-0.0993171, 0.0891037, 0.191597, 0.195382, 0.0999877, -0.0826916, -0.329877, -0.610744, -0.890273,
-1.13361, -1.3104, -1.39861, -1.38724, -1.27771, -1.08366, -0.829312, -0.546366, -0.270107}

```



O problema é que o plot como pontos (ListLinePlot) não escala o eixo X .

Só que se resolver por Plot[], como no [ActStudio é plotado?](#)

Parte do problema é que a largura do plot por pontos é definida pela lista de dados, e não por um scaling. Se estipularmos uma largura diferente para a discretização do modelo via scaling do **ts**, ele fica com largura diferente dos dados. Interpolar a lista de dados seria “feio”, mas talvez a única solução.

```
In[ ]:= Module[{dta, ts, dtapoints},
  dta = points1;
  ts = Range[0, (Length[dta] - 1) * 2, 2];
  Print[Length[ts]];
  Print[ts];
  dtapoints = Table[{i - 1, dta[[i]]}, {i, Range[1, Length[ts]]}];
  Print[dtapoints];
  mdl = Table[fmodelparam1[t, vpar1["M"], vpar1["τ"], vpar1["β"], vpar1["γ"]], {t, ts}];
  Print[mdl];
  Print[Plot[{dta, mdl}, ImageSize → Medium, {t, ...} < PlotRange → {{0, 200}, {-3, 3}}]];
];
```

Referências

- 1 <https://www.youtube.com/watch?v=29cVV sokRms&t=237s&list=WL&index=3>
- 2 ERIC - Least Squares Procedures