2º Trabalho Computacional - Análise Numérica (2020/21)

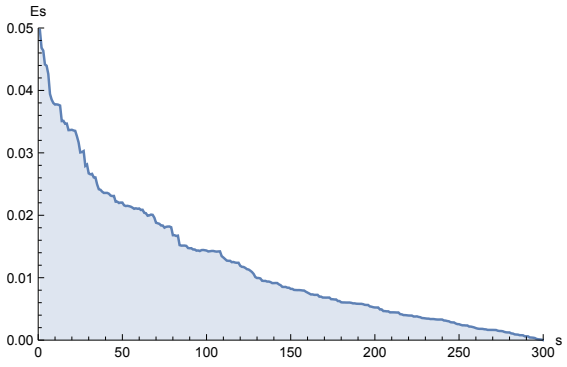
Relatório-resumo do Grupo: 18	**Nome:** Vasco Pearson (nº97015)

Questão 1): Nesta questão pretendemos calcular uma decomposição SVD da matriz $M \in [0,255]^{410\times300}$ associada ao mapa (preto-branco) da ilha das Flores. Com este objetivo em mente, foi implementado o método QR por Gram-Schmidt modificado (que reduz o efeito do cancelamento subtrativo em relação ao processo clássico de Gram-Schmidt), que levou à decomposição $M^TM = Q^TD^2Q$. Esta decomposição foi obtida chamando a função **DecompMTM**, que recebe uma matriz A e aplica o seguinte método, inicializando com $A_0 = M^TM$:

1. Aplica a decomposição QR chamando a função **DecompQR**, obtendo $A_i = Q_iR_i$
2. $A_{i+1} = R_iQ_i$
3. $\Theta_{i+1} = \Theta_iQ_i$ (Começamos com $\Theta_0 = Identidade$)

A função **SVDecomposition** calcula $U = MQ^TD^{-1}$ e retorna as matrizes U, D e Q que definem a decomposição singular $M = UDQ$. Considerou-se como condição de paragem 75 iterações do métodoQR e a diferença máxima entre a diagonal de A_n e A_{n-1} ser inferior a 200. Para finalizar, implementou-se a função **SVDecompositionCutted** que recebe s como parâmetro e devolve $M_s = U_{m\times s}D_{s\times s}Q_{s\times n}$

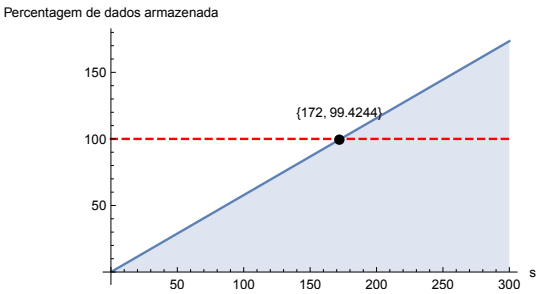
[R1a] Ao calcular $E_s = \frac{1}{mn}||M - M_s||_\infty$ em função de s, onde s representa o número de valores próprios da matriz M usados na reconstrução da imagem comprimida, pode-se verificar que E_s diminui conforme s aumenta, o que era de esperar, pois quando realizamos a decomposição $M_s = U_{m\times s}D_sQ_{s\times n}$ estamos a retirar informação. O erro aumenta exponencialmente quando retiramos muita informação, pois começamos a perder informação de valores próprios cada vez mais significantes e estes representam uma quantidade maior de informação sobre a imagem. Quando s se aproxima de n o erro tende para zero.



[R1b] Como se pode verificar, diminuindo o valor de s a imagem correspondente à matriz resultante vai perdendo informação, ficando cada vez mais comprimida e portanto a qualidade da imagem diminui, o que entra em concordância com o gráfico de E_s em função de s. A partir de um certo valor de s (neste caso entre $s = 173$ e $s = 300$) a imagem parece não sofrer alterações, isto é, existe um s limite para o qual a percentagem relativa à quantidade de pixeis precisa para armazenar a informação da imagem é maior que 100% e portanto tem-se que a compressão segundo o valor de s vai dar como output uma imagem muito semelhante à inicial, mesmo sendo menor o espaço ocupado por ela. Contudo, caso contrário não acontece, pois perde-se informação relevante da imagem. Esta percentagem é expressa por

$$\phi = \frac{(m+n+1)s}{mn} \times 100$$

.



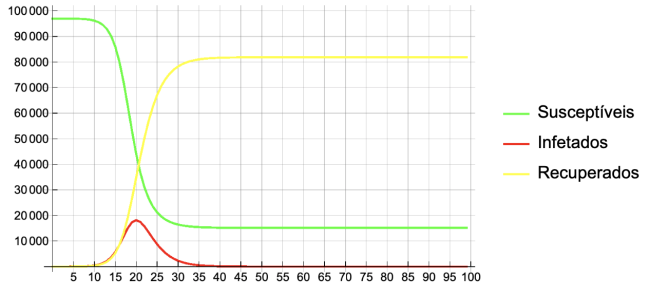
Em baixo apresentam-se as figuras obtidas por M_s para $s = 30$, $s = 172$ (valor do s limite) e $s = 300$ (não há compressão).

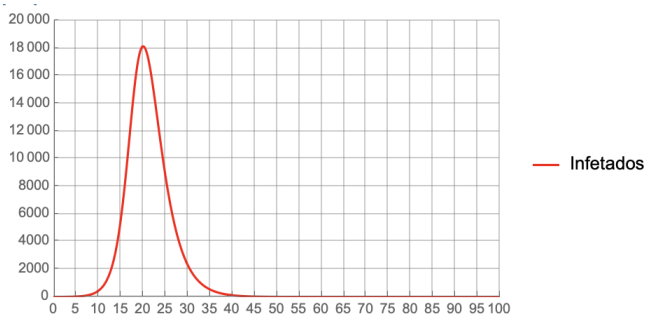
É visível que para $s = 172$, apesar de a percentagem de dados armazenados ser menor, a imagem é muito semelhante à original ($E_s = 0.0068207$). No caso de $s = 30$ a imagem apresenta um erro de $E_s = 0.0267365$ e por isso já não tem muita semelhança com a original.

Questão 2)

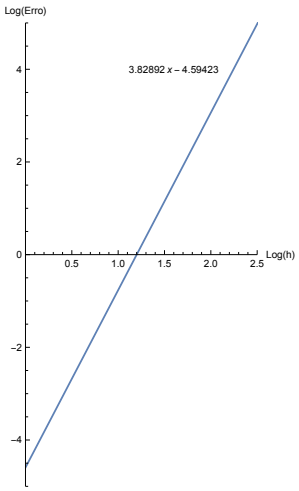
Métodos Multipasso são métodos utilizados para obter soluções numéricas de equações diferenciais ordinárias. Neste exercício vamos usar o método de Adams-Bashforth de ordem 4 com inicialização por Runge-Kutta, para resolver o sistema de equações diferencias ordinárias resultante do modelo SIR, apresentado no enunciado. A expressão do método de Adams-Bashforth de ordem 4 requer o conhecimento dos valores y_0, y_1, y_2, y_3, e daí inicializarmos com o método de Runge-Kutta de ordem 4.

[R2a] Foi simulado o resultado para $p = 97015$ e $R_0 = 1 + 18/15$ e obtiveram-se os gráficos abaixo. O primeiro corresponde ao número de susceptíveis, infetados e recuperados em função do tempo, enquanto que o segundo diz respeito apenas aos infetados.





[R2a] Analisou-se experimentalmente a ordem de convergência do método recorrendo à técnica dos mínimos quadrados. Dizemos que há convergência de ordem r se $e = y(t_n) - y_n = O(h^r)$, portanto temos que o erro e é da forma $e = C \times h^r$. De modo a aplicar a técnica dos mínimos quadrados fizemos a linearização $log(e) = log(C) + rlog(h)$. Usou-se o comando LinearSolve do Mathematica para resolver o sistema linear e obteve-se assim a ordem de convergência r correspondente ao declive da reta obtida.

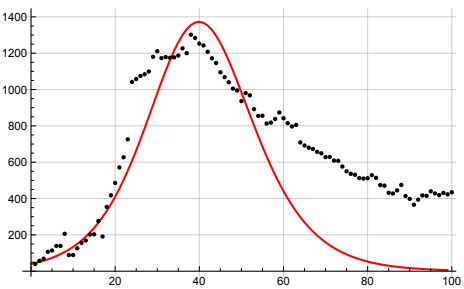


Teoricamente, a ordem de convergência é 4 para o método de Adams-Bashforth. O resultado obtido foi $r = 3.82892$ como se pode observar no gráfico à esquerda, o que não se afasta muito do resultado teórico.

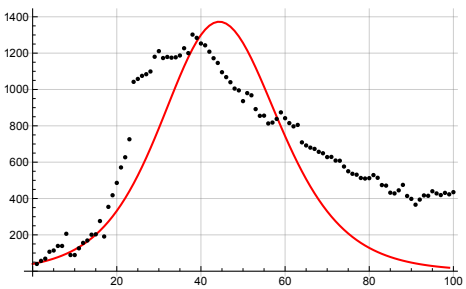
[R2b] Nesta alínea identificou-se o número de infetados com o número de internados do primeiro trabalho ($i(0) = 40$ em (i) e (ii), onde se usaram os dados do primeiro projeto; $i(0) = 317$ em (iii) onde se usaram os dados dos últimos 100 dias antes de Dezembro) e fixou-se a população em $p = 170000$. O objetivo da função **MelhorAjuste** é correr a função **AdamsBashforth** para vários valores de R_0 (mantendo a população, $s(0)$, $i(0)$, $r(0)$ e β constantes) e verificar qual destes valores minimiza a distância entre o número de infetados dado pelo modelo SIR e o número de internados dado pela DGS (pela *norma l2*). A função tem *RInicio* e *RFim* como argumentos, que definem entre que valores varia o R_0. Verificou-se que os valores de R_0 obtidos estavam todos entre 1 e 1.5, portanto estes foram os valores dados como argumento no código apresentado, de forma a otimizar o tempo do programa.

Na alínea (ii) foi desenvolvido uma função **MinBeta** que corre a função **MelhorAjuste** para vários valores de β (nos argumentos da função especificamos qual o valor inicial, final e o step). Esta parte do código apresenta-se comentada devido ao tempo que consome, apresentando-se apenas o valor final de β obtido para cada caso. Na alínea (iii) procedeu-se do mesmo modo que em (ii), apenas alterando a lista de valores da DGS e o número de infetados inicialmente.

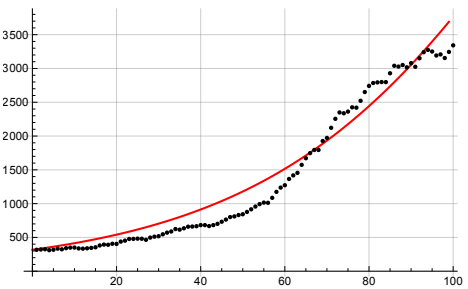
[R2b(i)] Usando a função **MelhorAjuste** com $\beta = 1$ obtemos:
$R = 1.14$
$\beta = 1$
Norma do vetor dos erros: 3035.27



[R2b(ii)] Variando β e R_0 simultaneamente, obtemos:
$R = 1.14$
$\beta = 0.9$
Norma do vetor dos erros: 2865.86



[R2b(iii)] Variando β e R_0 com os novos dados, obtemos:
$R = 1.43$
$\beta = 0.09$
Norma do vetor dos erros: 1998.27



Ao analisar os gráficos, verificou-se que o Modelo SIR não se ajusta aos dados da melhor forma, pois mesmo sendo menor o erro ao ajustar o valor de β, a norma do vetor dos erros usando os primeiros dados foi 2865.86. Obtemos a melhor aproximação para os valores dos últimos 100 dias antes de Dezembro, mesmo assim a norma do vetor dos erros ficou próxima de 2000.
O facto de os dados correspondem ao número de internados, e não de infetados pode justificar algumas irregularidades no ajuste do modelo. Para além disso, há outras condições que podem levar a um pior desempenho deste modelo, tais como o facto do R_0 variar com o tempo, a sociedade não ser fechada e a influência das medidas de prevenção tomadas pelo governo no número de infetados. Para uma previsão mais acertada teria-se de considerar mais fatores e consequentemente, um modelo mais complexo.

– Código Mathematica –

Questão 1

```
Clear["Global`*"];
link1 = "https://upload.wikimedia.org/wikipedia/";
link2 = "commons/1/16/Locator_map_Azores_Flores.png";
FloresMapa = Import[StringJoin[link1, link2]];
FloresMapa =
 ColorConvert[ImageResize[ImageTake[FloresMapa, {70, 911}, {485, 1100}], 300], GrayLevel];
M = ImageData[FloresMapa, "Byte"]; size = Dimensions[M];
(*Norma 2 de w.*)
norma2[w_] := Sqrt[Total[Map[#^2 &, w]]];
(*Gram-Shmidt modificado*)
ortonormalizacao[A_] := Module[{k, n, a, vk, e, uj, j},
  n = Length[A]; a = Transpose[A];
  vk = a[[1]]; e = Table[1, {i, n}];
  e[[1]] = vk/norma2[vk];
  For[k = 2, k <= n, k++,
   uj = a[[k]];
   For[j = 1, j <= k - 1, j++,
     uj = uj - Dot[a[[k]], e[[j]]]*e[[j]]];
   e[[k]] = N[uj/norma2[uj]]
   ];
  Transpose[e]];
(*Decomposicao QR*)
DecompQR[A_] := Module [{k, Ak, Qk, Rk},
  Ak = A; Qk = ortonormalizacao[Ak]; Rk = Dot[Transpose[Qk], Ak];
  {Qk, Chop[Rk]}
  ];
(*Decomposicao MTM=QTAQ; itermax é o numero de iterações max do metodo QR;
abserr é a diferença min entre valores proprios de duas matrizes A na sucessão de iterações do metodo QR;*)
DecompMTM[matrix_, size_, itermax_, abserr_] :=
 Module[{i, Aiter, Aiterb4, Q, Qiter, R, valerr},
  Aiter = matrix; valerr = 10^5;
  Q = IdentityMatrix[size[[2]]];
  For[i = 1, valerr > abserr && i < itermax, i++,
   Aiterb4 = Aiter;
   {Qiter, R} = DecompQR[Aiter];
   Aiter = Dot[R, Qiter];
   Q = Dot[Transpose[Qiter], Q];
   valerr = Max[Abs[Diagonal[Aiter] - Diagonal[Aiterb4]]];
   ];
  Return[List[Transpose[Q], Chop[Aiter], Q]]
  ];
(*Decomposicao SVD MTM=UDQ*)
SVDecomposition[matrix_, size_, itermax_, abserr_] :=
 Module[{MTM, QT, DSquare, Q, UF, U, DF, D, QF},
```

```
  MTM = Dot[Transpose[matrix], matrix];
  {QT, DSquare, QF} = DecompMTM[MTM, size, itermax, abserr];
  DF = DiagonalMatrix[Sqrt[Diagonal[DSquare]]];
  UF = Dot[matrix, QT, Inverse[DF]];
  Return[List[UF, DF, QF]]
  ];
{MatrixUF1, MatrixDF1, MatrixQF1} = SVDecomposition[M, size, 75, 200];
(*matriz reconstruida com o fator s*)
SVDecompositionCutted[s_, size_] := Module[{Matrix},
  Matrix =
   Dot[MatrixUF1[[1 ;; size[[1]], 1 ;; s]], MatrixDF1[[1 ;; s, 1 ;; s]],
   MatrixQF1[[1 ;; s, 1 ;; size[[2]]]]];
  Return[Matrix]
  ];
ErroDecomposition[s_, size_, Matriz_] := Module[{Mgray, Erro},
  Mgray = SVDecompositionCutted[s, size];
  Erro = (1/(size[[1]]*size[[2]]))*(Norm[Matriz - Mgray, Infinity])
  ];
(*1a*)
DiscretePlot[ErroDecomposition[s, size, M], {s, 1, 300},
 PlotRange -> {{0, 300}, {0, 0.05}}, AxesLabel -> {"s", "Es"}]\[NewLine]
(*1b*)
Image[SVDecompositionCutted[30, size], "Byte"]\[NewLine]
ErroDecomposition[30, size, M]\[NewLine]
Image[SVDecompositionCutted[172, size], "Byte"]\[NewLine]
ErroDecomposition[172, size, M]\[NewLine]
Image[SVDecompositionCutted[300, size], "Byte"]\[NewLine]
SavedPixels1[
  s_] := (((size[[1]] + size[[2]])*s) + s)*100/(size[[1]]*size[[2]]);
Show[DiscretePlot[SavedPixels1[s], {s, 1, size[[2]]},
  AxesLabel -> {"s", "Percentagem de dados armazenada" },
  Epilog -> {PointSize[Large], Black,
    Point[{172, SavedPixels1[172] }],
    Text[{172, N[SavedPixels1[172]]}, {172, 120}]}],
 Plot[100, {s, 0, 300}, PlotStyle -> {Red, Dashed}]]
```

Questão 2

```
Clear["Global`*"];
p1 = 97015;
slinha = N[Function[{s, i, Beta}, -s*i*Beta]];
ilinha = N[Function[{R0, s, i, Beta}, (R0*s - 1)*i*Beta/R0]];
AdamsBashforth[x_, y_, z_, ratio_, Beta_, ds_, di_, h_, d_] :=
  Module[{s = x, i = y, r = z, R = ratio, B = Beta, t = 0, Fs0, Fi0,
    Fs1, Fi1, Fs2, Fi2, Fs3, Fi3, ys1, yi1, ys2, yi2, ys3, yi3,
    dslist = {ds[x, y, Beta]}, dilist = {di[ratio, x, y, Beta]},
    dias = d, j, tdata = {0}, sdata = {x}, rdata = {z}, idata = {y},
    l},
   For[l = 1, l <= 3, l++,
    Fs0 = ds[s, i, B];
    Fi0 = di[R, s, i, B];
    Fs1 = ds[s + h*Fs0/2, i + h*Fi0/2, B];
    Fi1 = di[R, s + h*Fs0/2, i + h*Fi0/2, B];
    Fs2 = ds[s + h*Fs1/2, i + h*Fi1/2, B];
    Fi2 = di[R, s + h*Fs1/2, i + h*Fi1/2, B];
    Fs3 = ds[s + h*Fs2, i + h*Fi2, B];
    Fi3 = di[R, s + h*Fs2, i + h*Fi2, B];
    t = t + h;
    s = s + h*(Fs0 + 2*Fs1 + 2*Fs2 + Fs3)/6;
```

```
    i = i + h*(Fi0 + 2*Fi1 + 2*Fi2 + Fi3)/6;
    AppendTo[tdata, t];
    AppendTo[sdata, s]; AppendTo[idata, i]; AppendTo[rdata, 1 - i - s];
    AppendTo[dslist, ds[s, i, B]]; AppendTo[dilist, di[R, s, i, B]];
    ];
   t = t + h;
   For[j = 1, t + h <= dias, t = t + h,
    s = s +
      h*((55/24)*dslist[[j + 3]] - (59/24)*dslist[[j + 2]] + (37/24)*
          dslist[[j + 1]] - (9/24)*dslist[[j]]);
    i = i +
      h*((55/24)*dilist[[j + 3]] - (59/24)*dilist[[j + 2]] + (37/24)*
          dilist[[j + 1]] - (9/24)*dilist[[j]]);
    AppendTo[dslist, ds[s, i, B]]; AppendTo[dilist, di[R, s, i, B]];
    AppendTo[tdata, t]; AppendTo[sdata, s]; AppendTo[idata, i];
    AppendTo[rdata, 1 - i - s];
    j++;
    ];
   Return[{tdata, sdata, idata, rdata}]
   ;];
(*2a*)
lista = AdamsBashforth[1-2/p1,2/p1,0,1+18/15,1,slinha,ilinha,0.1,100];
ListLinePlot[{Table[{lista[[1]][[i]],p1*lista[[2]][[i]]},{i,1,Length[lista[[2]]],1/0.1}],
  Table[{lista[[1]][[i]],p1*lista[[3]][[i]]},{i,1,Length[lista[[3]]],1/0.1}],
  Table[{lista[[1]][[i]],p1*lista[[4]][[i]]},{i,1,Length[lista[[4]]],1/0.1}]},
 Ticks->{Table[5*i,{i,0,20}],Table[10000*i,{i,0,18}]},
 GridLines->{Table[5*i,{i,0,20}],Table[10000*i,{i,0,18}]},
 PlotStyle->{Green,Red,Yellow},
 PlotLegends->{"Susceptíveis","Infetados","Recuperados"}]\[NewLine]
ListLinePlot[
 Table[{lista[[1]][[i]],p1*lista[[3]][[i]]},{i,1,Length[lista[[3]]],1/1}],
 Ticks->{Table[5*i,{i,0,20}],Table[2000*i,{i,0,20}]},
 GridLines->{Table[5*i,{i,0,20}],Table[2000*i,{i,0,20}]},
 PlotStyle->Red,PlotLegends->{"Infetados"},
 PlotRange->{{0,100},{0,20000}}]\[NewLine]
norma2[w_] := Sqrt[Total[Map[#^2 &, w]]];
prev1 = AdamsBashforth[1 - 2/p1, 2/p1, 0, 11/5, 1, slinha, ilinha, 1, 100][[3]];
prev2i = AdamsBashforth[1 - 2/p1, 2/p1, 0, 11/5, 1, slinha, ilinha, 0.5, 100][[3]];
prev2 = Table[prev2i[[i]], {i, 1, Length[prev2i], 1/0.5}];
prev3i = AdamsBashforth[1 - 2/p1, 2/p1, 0, 11/5, 1, slinha, ilinha, 0.01, 100][[3]];
prev3 = Table[prev3i[[i]], {i, 1, Length[prev3i], 1/0.01}];
reali = AdamsBashforth[1 - 2/p1, 2/p1, 0, 11/5, 1, slinha, ilinha, 0.005, 100][[3]];
real = Table[reali[[i]], {i, 1, Length[reali], 1/0.005}];
Convergencia[P1_, P2_, P3_, PR_, h1_, h2_, h3_, hR_] :=
  Module[{E1 = P1, E2 = P2, E3 = P3, ER = PR, Lh, LE },
   E1 = Log[norma2[ER - E1]];
   E2 = Log[norma2[ER - E2]];
   E3 = Log[norma2[ER - E3]];
   Lh = Log[List[h1, h2, h3]];
   LE = List[E1, E2, E3];
   LinearSolve[{{Length[Lh], Total[Lh]}, {Total[Lh], Total[Lh^2]}}, {Total[LE], Lh.LE}]];
{logC, r} = Convergencia[prev1, prev2, prev3, real, 1, 0.5, 0.01, 0.005]\[NewLine]
Plot[logC + r*x, {x, 0, 5}, AxesLabel -> {"Log(h)", "Log(Erro)"},
 PlotRange -> {{0, 2.5}, {-5, 5}}, AspectRatio -> 2,
 Epilog -> {Text[-4.59423 + 3.82892 x, {1.6, 4}]}]\[NewLine]
(*2b*)
listaValoresY = {40, 57, 69, 107, 114, 139, 139, 206, 89, 89, 126,
   156, 169, 201, 203, 276, 191, 354, 418, 486, 571, 627, 726, 1042,
```

```
   1058, 1075, 1084, 1099, 1180, 1211, 1173, 1179, 1175, 1177, 1187,
   1227, 1200, 1302, 1284, 1253, 1243, 1208, 1172, 1146, 1095, 1068,
   1040, 1005, 995, 936, 980, 968, 892, 855, 856, 813, 818, 838, 874,
   842, 815, 797, 805, 709, 692, 680, 673, 657, 649, 628, 629, 609,
   608, 576, 550, 536, 531, 513, 510, 512, 529, 514, 474, 471, 432,
   428, 445, 475, 414, 398, 366, 394, 417, 415, 440, 428, 419, 431,
   423, 435};
listaValores = {};
For[i = 1, i <= Length[listaValoresY], i++,
 AppendTo[listaValores, {i, listaValoresY[[i]]}]];
MelhorAjuste[populacao_, susceptiveis_, dias_, valoresReais_,
   RInicio_, RFim_, Beta_, perg_] :=
  Module[{p1 = populacao, d = dias, s0 = susceptiveis, Rmin,
    erros = {}, B = Beta, errosMin = {10^50}, lista1E1, listaE1,
    listaE2, e},
   For[R = RInicio, R <= RFim, R = R + 0.01,
    erros = {};
    lista1E1 =
     AdamsBashforth[s0/p1, (p1 - s0)/p1, 0, R, B, slinha, ilinha, 0.1,
        d][[3]];
    listaE1 = Table[lista1E1[[i]], {i, 1, Length[lista1E1], 1/0.1}];
    erros = Abs[p1*listaE1 - valoresReais[[All, 2]]];
    If [N[norma2[erros]] < N[norma2[errosMin]], errosMin = erros;
     Rmin = R;];
    ];
   If[perg == "2ii", norma2[errosMin],
    Print["R = ", Rmin];
    Print["\[Beta] = ", B];
    Print["Norma do vetor dos erros: ", norma2[errosMin]];
    listaE2 =
     AdamsBashforth[s0/p1, (p1 - s0)/p1, 0, Rmin, B, slinha, ilinha,
      0.1, d];
    ListLinePlot[
     Table[{listaE2[[1]][[i]], p1*listaE2[[3]][[i]]}, {i, 1,
       Length[listaE2[[3]]], 1/0.1}], Ticks -> Automatic,
     GridLines -> Automatic, PlotStyle -> Red,
     Epilog -> {PointSize[0.01], Black, Map[Point, valoresReais]}]]];
p2 = 170000;
(*i*)
MelhorAjuste[p2,p2-40,100,listaValores,1,1.5,1,""]\[NewLine]
(*ii*)
MinBeta[ini_, fin_, stp_, pop_, i_, lis_] :=
  Module[{nMin = 10^10, best, k, n},
   For[k = ini, k <= fin, k = k + stp,
    n = MelhorAjuste[pop,pop-i,100,lis,1,1.5,k,"2ii"];
    If[n < nMin, nMin = n; best = k]];
   best];
(*MinBeta[0,2,0.5,p2,40,listaValores]\[NewLine]*)
(*MinBeta[0.6,1.4,0.1,p2,40,listaValores]\[NewLine]*)
(*MinBeta[0.8,1,0.05,p2,40,listaValores]\[NewLine]*)
(*MinBeta[0.85,0.95,0.01,p2,40,listaValores]\[NewLine]*)
MelhorAjuste[p2,p2-40,100,listaValores,1,1.50,0.90,""]\[NewLine]
(*iii*)
ultimosCem = {317, 321, 325, 311, 317, 334, 324, 341, 349, 350, 337,
   334, 339, 345, 354, 381, 394, 391, 406, 404, 438, 452, 477, 478,
   482, 480, 465, 497, 511, 518, 546, 571, 588, 624, 615, 635, 659,
   661, 666, 682, 682, 668, 682, 701, 732, 764, 801, 811, 831, 843,
   877, 916, 957, 993, 1015, 1012, 1086, 1174, 1237, 1272, 1365,
```

```
   1418, 1455, 1574, 1672, 1747, 1794, 1794, 1927, 1972, 2122, 2255,
   2349, 2337, 2362, 2425, 2420, 2522, 2651, 2742, 2785, 2794, 2799,
   2798, 2929, 3040, 3028, 3051, 3017, 3079, 3025, 3151, 3241, 3275,
   3251, 3192, 3208, 3155, 3245, 3342};
listaVal2 = {};
For[i = 1, i <= Length[ultimosCem], i++,
  AppendTo[listaVal2, {i, ultimosCem[[i]]}]];
(*MinBeta[0,2,0.5,p2,listaVal2]\[NewLine]*)
(*MinBeta[0,0.9,0.1,p2,317,listaVal2]\[NewLine]*)
(*MinBeta[0,0.2,0.01,p2,317,listaVal2]\[NewLine]*)
MelhorAjuste[p2,p2-317,100,listaVal2,1,1.5,0.090,""]
```

Observação 1: Devido ao tamanho ocupado pelos dados, foi utilizada esta página para finalizar o relatório.
Observação 2: O código todo demora menos de um minuto a correr.