INTRODUÇÃO

O problema consiste em aproximar um modelo matemático por duas funções utilizando o método dos mínimos quadrados e compará-las para saber qual se adequa mais aos resultados esperados. São conhecidos 101 pontos desse modelo.

DESENVOLVIMENTO

O gráfico de dispersão dos pontos conhecidos é:

Percebe-se que o gráfico pode ser aproximado por uma função exponencial:

f(x) ≈ g(x) = a*e^(b*x). Linearizando: ln(f(x)) ≈ ln(g(x)) = ln(a*e^(b*x)) = lna + b*x. Portanto, ln(f(x)) ≈ ln(g(x)) = lna + b*x, em que F(X) = ln(f(x)) e G(X) = ln(g(x)), de forma que F(X) ≈ G(X) = lna + b*x é o novo problema.

Assim, a1 = lna, g1(x) = 1, a2 = b, g2(x) = x.

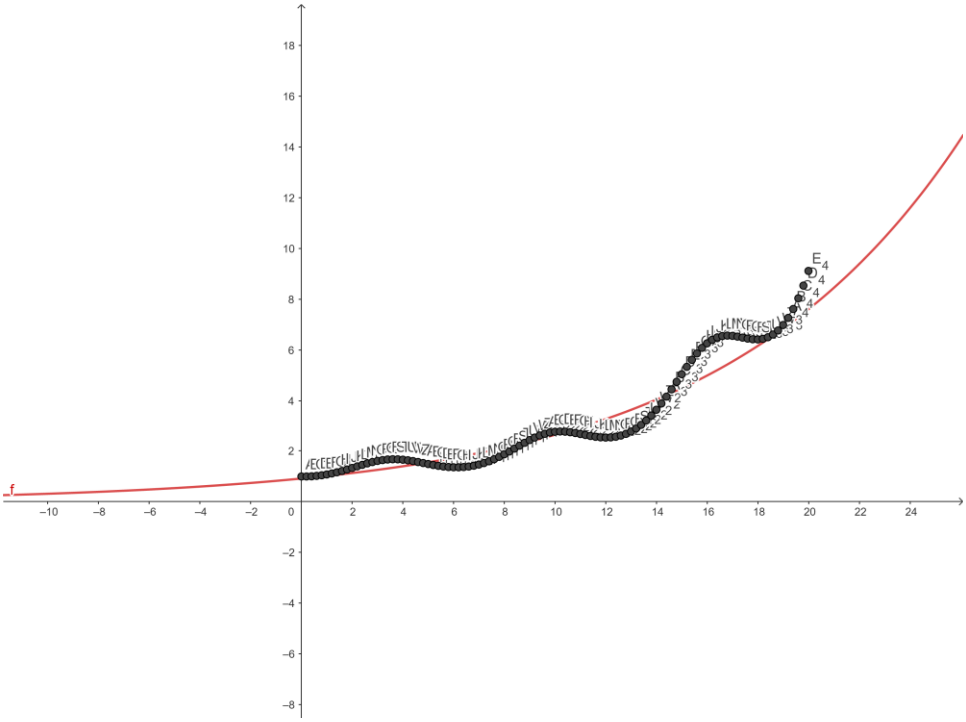
Utilizando um programa em C para ler os pontos e aplicar o método dos mínimos quadrados,

```c
void getAnswer1(double *vector_t, double *vector_y) {
    //Realizando os somatórios necessários:
    double sum1 = MAX_POINTS;

    double sumXi = 0;
    for(int i = 0; i < MAX_POINTS; i++)
        sumXi += vector_t[i];

    double sumXi2 = 0;
    for(int i = 0; i < MAX_POINTS; i++)
        sumXi2 += vector_t[i] * vector_t[i];

    //getLnVector para encontrar F(Xi) = ln(f(xi)) = ln(vector_y[xi])
    double* LnVector = getLnVector(vector_y);
    double sumFi = 0;
    for(int i = 0; i < MAX_POINTS; i++)
        sumFi += LnVector[i];

    double sumXiFi = 0;
    for(int i = 0; i < MAX_POINTS; i++)
        sumXiFi += vector_t[i] * LnVector[i];

    double denominator = sum1 * sumXi2 - sumXi * sumXi;
    double a = (sumFi * sumXi2 - sumXi * sumXiFi)/denominator;
    double b = (sum1 * sumXiFi - sumXi * sumFi)/denominator;

    printf("A funcao encontrada e: %.8lf * e^(%.8lf * x)\n", exp(a), b);
    free(LnVector);
}
```
,

tem-se o resultado da g:

```
C:\Windows\system32\cmd.exe
A funcao encontrada e: 0.92010736 * e^(0.10551507 * x)

Pressione qualquer tecla para continuar. . .
```
.

Ao plotar essa função encontrada juntamente com o gráfico dos pontos conhecidos do modelo matemático em questão, obtemos



O cálculo da soma dos resíduos ao quadrado para essa aproximação é calculado em F e G, e serve como boa aproximação para f e g:

```c
void residualsSquared(double *vector_t, double *vector_y) {
    double* F = getLnVector(vector_y);

    double g[MAX_POINTS];
    for(int i = 0; i < MAX_POINTS; i++)
        g[i] = 0.92010736 * pow(E, 0.10551507 * vector_t[i]); //vem de getAnswer1

    double* G = getLnVector(g);

    double result = 0;
    for(int i = 0; i < MAX_POINTS; i++)
        result += pow(F[i] - G[i], 2);

    free(F);
    free(G);

    printf("A soma dos residuos ao quadrado de (F - G) e: %.8lf\n", result);
}
```

```
C:\Windows\system32\cmd.exe
A soma dos residuos ao quadrado de (F - G) e: 2.68377863

Pressione qualquer tecla para continuar. . .
```

Aparentemente, pelo gráfico, a escolha da função exponencial para a aproximação parece ter sido correta, o que se confirma pelo cálculo do R2:

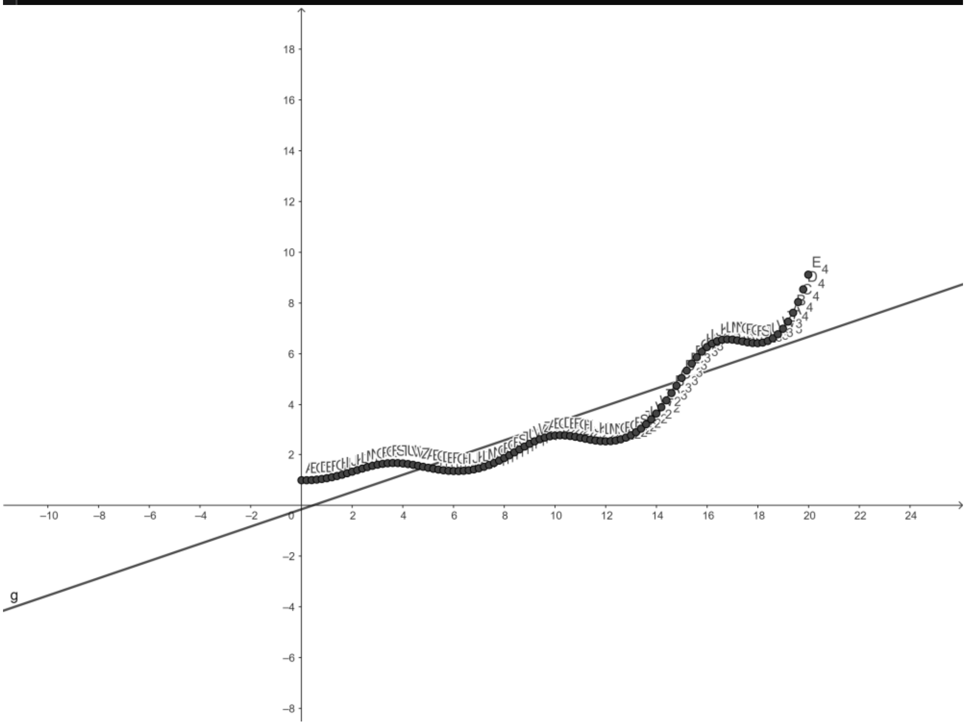
```c
void R2One(double *vector_t, double *vector_y) {
    double SQRes = 0;
    double g[MAX_POINTS];
    for(int i = 0; i < MAX_POINTS; i++)
        g[i] = 0.92010736 * pow(E, 0.10551507 * vector_t[i]);
    for(int i = 0; i < MAX_POINTS; i++)
        SQRes += pow(vector_y[i] - g[i], 2);

    double y_ = 0;
    double fxiSum = 0;
    for(int i = 0; i < MAX_POINTS; i++)
        fxiSum += vector_y[i];
    y_ = fxiSum / MAX_POINTS;

    double SQtot = 0;
    for(int i = 0; i < MAX_POINTS; i++)
        SQtot += pow(vector_y[i] - y_, 2);

    double R2 = 1 - SQRes / SQtot;
    printf("O coeficiente R2 para essa aproximacao e: %.8lf", R2);
}
```

```
C:\Windows\system32\cmd.exe
O coeficiente R2 para essa aproximacao e: 0.93633775
Pressione qualquer tecla para continuar. . .
```

e, quanto mais próximo de 1 o resultado do R2, melhor. Então, a função escolhida para a aproximação é muito boa.

Repetindo o mesmo processo para a aproximação por uma função linear do tipo a + bx, obtemos os resultados:

```
A funcao encontrada e: -0.14386711 + 0.33974452 * x
A soma dos residuos ao quadrado de (f - g) e: 78.37854802
O coeficiente R2 para essa aproximacao e: 0.83490654
Pressione qualquer tecla para continuar. . .
```



<u>CONCLUSÃO</u>

Levando em consideração dois fatos:

<u>1</u> - A soma dos resíduos ao quadrado da aproximação por uma função exponencial é menor do que pela aproximação por uma função linear, isto é, o erro residual é menor.

<u>2</u> – O R2 da aproximação pela função exponencial é mais próximo de 1 do que pela função linear.

<u>Assim, conclui-se que o ajuste por uma **função exponencial** é o melhor para esse modelo matemático.</u>

O código completo deste trabalho pode ser encontrado em https://github.com/h-Soares/USP-repository/blob/main/CN%20W2/mmq.c