MATH303-22S2 - *Assignment 1*

`Due Date` : **Monday 19th September 5pm**

Submission by: Richard Hodges, Henry Hastings

Student IDs: 11139318, 35556669

Code is written in Julia and presented in a Pluto.jl notebook.

```
using LinearAlgebra
```

Question One

$$A = \begin{bmatrix} 3 & 4 \\ 0 & -2 \\ 0 & 1 \\ 0 & 2 \end{bmatrix}$$

a) Householder Reflectors:

Since column 1 is correctly formatted we only need to perform one Householder reflection to diagonalize this matrix. We take the portion of the matrix we will be acting upon and add padding. This results in:

$$\vec{m} = \begin{bmatrix} 0 \\ -2 \\ 1 \\ 2 \end{bmatrix}$$

We must find $\theta = ||\vec{m}||_2$ and subtract $\theta \times e_2$ from our $\vec{m}$:

$$\theta = ||\vec{m}||_2 = 3$$

$$\vec{v} = \vec{m} - \theta * e_2 = \begin{bmatrix} 0 \\ -5 \\ 1 \\ 2 \end{bmatrix}$$

We then find

$$\beta = \frac{2}{\vec{v}^T \vec{v}}$$

We use the following formula to find $\boldsymbol{H}$:

$$H = I - \beta \vec{v}\vec{v}^T = \frac{1}{3}\begin{bmatrix} 3 & 0 & 0 & 0 \\ 0 & -2 & 1 & 2 \\ 0 & 1 & 2.8 & -0.4 \\ 0 & 2 & -0.4 & 2.2 \end{bmatrix}$$

Thus, noting there is only one householder matrix used in this transformation, we find:

$$Q = H = \frac{1}{3}\begin{bmatrix} 3 & 0 & 0 & 0 \\ 0 & -2 & 1 & 2 \\ 0 & 1 & 2.8 & -0.4 \\ 0 & 2 & -0.4 & 2.2 \end{bmatrix} \quad R = QA = \begin{bmatrix} 3 & 4 \\ 0 & 3 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$$

---

b) Givens Rotations:

We first must find $\boldsymbol{c}$ and $\boldsymbol{s}$ for row 3 column 2 of A using the given rule:

$$c = \frac{a_{22}}{\sqrt{a_{22}^2 + a_{32}^2}} = \frac{-2}{\sqrt{5}}, \quad s = \frac{a_{32}}{\sqrt{a_{22}^2 + a_{32}^2}} = \frac{1}{\sqrt{5}}$$

We create our Givens rotation matrix like so:

$$G_{32} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c & s & 0 \\ 0 & -s & c & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

This results in our first Givens matrix:

$$G_{32} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{-2}{\sqrt{5}} & \frac{1}{\sqrt{5}} & 0 \\ 0 & \frac{-1}{\sqrt{5}} & \frac{-2}{\sqrt{5}} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \Rightarrow \quad G_{32}A = \begin{bmatrix} 3 & 4 \\ 0 & \sqrt{5} \\ 0 & 0 \\ 0 & 2 \end{bmatrix}$$

We repeat the process with row 4 column 2, using $\boldsymbol{G_{32}A}$ as our new "$\boldsymbol{A}$":

$$c = \frac{a_{22}}{\sqrt{a_{22}^2 + a_{42}^2}} = \frac{\sqrt{5}}{3}, \quad s = \frac{a_{42}}{\sqrt{a_{22}^2 + a_{42}^2}} = \frac{2}{3}$$

$$\Rightarrow \quad G_{42} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{\sqrt{5}}{3} & 0 & \frac{2}{3} \\ 0 & 0 & 1 & 0 \\ 0 & -\frac{2}{3} & 0 & \frac{\sqrt{5}}{3} \end{bmatrix} \quad \Rightarrow \quad G_{42}G_{32}A = \begin{bmatrix} 3 & 4 \\ 0 & 3 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} = R$$

We obtain Q in the following way:

$$Q = G_{32}^T G_{42}^T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -\frac{2}{3} & \frac{-1}{\sqrt{5}} & \frac{4}{3\sqrt{5}} \\ 0 & \frac{1}{3} & \frac{-2}{\sqrt{5}} & \frac{-2}{3\sqrt{5}} \\ 0 & \frac{2}{3} & 0 & \frac{\sqrt{5}}{3} \end{bmatrix}$$

Question Two

As $A\vec{x} = \vec{b}$ is an underdetermined system, following the notes in section 4.1.3 we must find the economy QR factorisation of A^T, then solve $\hat{R}^T\vec{u} = \vec{b}$ for $\vec{u}$ and then finally set $\vec{x} = Y\vec{u}$ to find the LSS for $A\vec{x} = \vec{b}$.

Given the QR factors of A^T we can find its economy QR factorisation by trimming off the row of zeros in R and then removing the necessary column in Q:

$$A^T = Y\hat{R} = \frac{1}{3}\begin{bmatrix} 2 & -2 \\ 1 & 2 \\ 2 & 1 \end{bmatrix}\begin{bmatrix} 1 & 2 \\ 0 & 1 \end{bmatrix} = \frac{1}{3}\begin{bmatrix} 2 & 2 \\ 1 & 4 \\ 2 & 5 \end{bmatrix}$$

This gives us the following expression to evaluate, following our steps

$$\hat{R}^T\vec{u} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

We can solve this system using forward substitution

$$\begin{bmatrix} 1 & 0 & : & 1 \\ 2 & 1 & : & 1 \end{bmatrix} \quad \Rightarrow \quad u_1 = 1, \quad u_2 = -1 \quad \Rightarrow \quad \vec{u} = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$$

We use this $\vec{u}$ to find $\vec{x}$ continuing our steps:

$$Y\vec{u} = \vec{x} \quad \Rightarrow \quad \frac{1}{3}\begin{bmatrix} 2 & -2 \\ 1 & 2 \\ 2 & 1 \end{bmatrix}\begin{bmatrix} 1 \\ -1 \end{bmatrix} = \vec{x} \quad \Rightarrow \quad \vec{x} = \frac{1}{3}\begin{bmatrix} 4 \\ -1 \\ 1 \end{bmatrix}$$

---

b) We can find the general solution to the equation $A\vec{x} = \vec{b}$ by doing Gaussian elimination on the system, noting we already know A from our expression of A^T

$$\frac{1}{3}\begin{bmatrix} 2 & 1 & 2 \\ 2 & 4 & 5 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \quad \Leftrightarrow \quad \begin{bmatrix} 2 & 1 & 2 & : & 3 \\ 2 & 4 & 5 & : & 3 \end{bmatrix} \quad \Rightarrow \quad \begin{bmatrix} 2 & 1 & 2 & : & 3 \\ 0 & 3 & 3 & : & 0 \end{bmatrix}$$

$$\Rightarrow \quad x_1 = \frac{3-t}{2}, \quad x_2 = -t, \quad x_3 = t$$

A particular solution to this equation could be

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} \frac{3}{2} \\ 0 \\ 0 \end{bmatrix}$$

Question 3

We are given:

$$B = A + \vec{u}\vec{v}^T$$

$$B^{-1} = A^{-1} - \alpha A^{-1}\vec{u}\vec{v}^T A^{-1}$$

Thus we must show:

$$I = (A + \vec{u}\vec{v}^T)(A^{-1} - \alpha A^{-1}\vec{u}\vec{v}^T A^{-1})$$

Giving the following:

$$I = I + \vec{u}\vec{v}^T A^{-1} - \alpha\vec{u}\vec{v}^T A^{-1} - \alpha\vec{u}\vec{v}^T A^{-1}\vec{u}\vec{v}^T A^{-1}$$

$$\alpha\vec{u}\vec{v}^T A^{-1} + \alpha\vec{u}\vec{v}^T A^{-1}\vec{u}\vec{v}^T A^{-1} = \vec{u}\vec{v}^T A^{-1}$$

$$\alpha\vec{u}\vec{v}^T + \alpha\vec{u}\vec{v}^T A^{-1}\vec{u}\vec{v}^T = \vec{u}\vec{v}^T$$

$$\alpha\vec{u}\vec{v}^T + \alpha\vec{u}(\vec{v}^T A^{-1}\vec{u})\vec{v}^T = \vec{u}\vec{v}^T$$

$$\alpha\vec{u}\vec{v}^T + \alpha\vec{u}\vec{v}^T(\vec{v}^T A^{-1}\vec{u}) = \vec{u}\vec{v}^T$$

$$\vec{u}\vec{v}^T(\alpha + \alpha\vec{v}^T A^{-1}\vec{u}) = \vec{u}\vec{v}^T$$

We note that $\vec{v}^T A^{-1}\vec{u}$, and thus $\alpha + \alpha\vec{v}^T A^{-1}\vec{u}$ is a scalar meaning

$$\alpha + \alpha\vec{v}^T A^{-1}\vec{u} = 1$$

Rearranging for α:

$$\alpha = \frac{1}{1 + \vec{v}^T A^{-1}\vec{u}}$$

Therefore we have a value for α which satisfies $BB^{-1} = I$ making the statement for B^{-1} true.

Therefore:

$$B^{-1} = A^{-1} - \alpha A^{-1}\vec{u}\vec{v}^T A^{-1}$$

With:

$$\alpha = \frac{1}{1 + \vec{v}^T A^{-1}\vec{u}}$$

Question 4

a) Going with the eigenvalue route

$$det(A - \lambda I) \quad = \quad det \begin{bmatrix} 1-\lambda & 1 & 0 \\ 1 & a-\lambda & -2 \\ 0 & -2 & 10-\lambda \end{bmatrix}$$

Gives us the characteristic polynomial

$$P(\lambda) = (1-\lambda)((a-\lambda)(10-\lambda)-4)-(10-\lambda)$$

$$P(\lambda) = -\lambda^3 + 11\lambda^2 + a\lambda^2 - 11\lambda a - 5\lambda + 10a - 14$$

We set $P(\lambda) = 0$ to find λ the eigenvalues of A

$$-\lambda^3 + 11\lambda^2 + a\lambda^2 - 11\lambda a - 5\lambda + 10a - 14 = 0$$

We can set λ to 0, then solve for a

$$10a - 14 = 0 \quad \Rightarrow \quad a = \frac{7}{5}$$

This means there is only one solution for a ($\frac{7}{5}$) that gives us an eigen value of 0. The cubic $P(\lambda)$ for any $a \in \mathbb{R}$ is smooth and continuous. This tells us that the roots to the characteristic polynomial can change sign, but it can only happen when $a = \frac{7}{5}$.

We can solve again when $a \in \{1, 2\}$

$$a = 1 \;\Rightarrow\; \lambda \approx -0.21,\ 1.76,\ 10.43$$

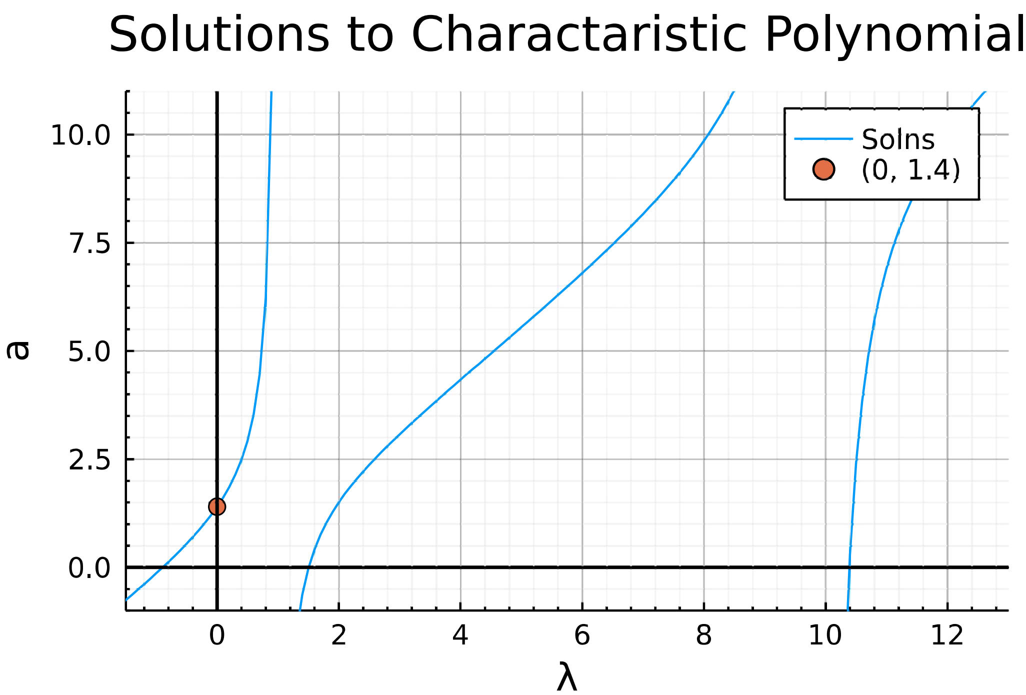
$$a = 2 \;\Rightarrow\; \lambda \approx 0.252,\ 2.27,\ 10.48$$

So we can see when $a > \frac{7}{5}$ we get positive eigenvalues and thus a positive definite matrix. $\square$

Supplementarily we can make a the subject of the system and plot the solution of for a with respect to λ. A little algebra yields:

$$a = \frac{\lambda^3 - 11\lambda^2 + 5\lambda + 14}{(\lambda-1)(\lambda-10)}$$

This has vertical asymptotes at $\lambda = 1$ and $\lambda = 10$. We can plot this as well. For every value of a we can draw a horizontal line and get the 3 roots of our characteristic polynomial, the eigenvalues of A. By observation it is clear the only time all 3 eigenvalues of A are positive is when $a > \frac{7}{5}$.



b) A is positive semi-definite when $a \geq \frac{7}{5}$, as this is when $\lambda \geq 0$ for all three eigenvalues of A.

c) When $a = 5$ we get the matrix

$$A = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 5 & -2 \\ 0 & -2 & 10 \end{bmatrix}$$

We make this matrix upper diagonal through row reduction

$$A \xrightarrow{R_2 = R_2 - R_1} \begin{bmatrix} 1 & 1 & 0 \\ 0 & 4 & -2 \\ 0 & -2 & 10 \end{bmatrix} \xrightarrow{R_3 = R_3 + \frac{1}{2}R_2} \begin{bmatrix} 1 & 1 & 0 \\ 0 & 4 & -2 \\ 0 & 0 & 9 \end{bmatrix} = U$$

We can find the inverse of U by using row reduction to make U I and doing the same operations on I

$$\begin{bmatrix} 1 & 1 & 0 & : & 1 & 0 & 0 \\ 0 & 4 & -2 & : & 0 & 1 & 0 \\ 0 & 0 & 9 & : & 0 & 0 & 1 \end{bmatrix} \Rightarrow \begin{bmatrix} 1 & 1 & 0 & : & 1 & 0 & 0 \\ 0 & 4 & -2 & : & 0 & 1 & 0 \\ 0 & 0 & 1 & : & 0 & 0 & \frac{1}{9} \end{bmatrix}$$

$$\Rightarrow \begin{bmatrix} 1 & 1 & 0 & : & 1 & 0 & 0 \\ 0 & 1 & 0 & : & 0 & \frac{1}{4} & \frac{1}{18} \\ 0 & 0 & 1 & : & 0 & 0 & \frac{1}{9} \end{bmatrix} \Rightarrow \begin{bmatrix} 1 & 0 & 0 & : & 1 & \frac{-1}{4} & \frac{-1}{18} \\ 0 & 1 & 0 & : & 0 & \frac{1}{4} & \frac{1}{18} \\ 0 & 0 & 1 & : & 0 & 0 & \frac{1}{9} \end{bmatrix}$$

Thus:

$$U^{-1} = \begin{bmatrix} 1 & \frac{-1}{4} & \frac{-1}{18} \\ 0 & \frac{1}{4} & \frac{1}{18} \\ 0 & 0 & \frac{1}{9} \end{bmatrix}$$

Multiplying $\boldsymbol{U^{-1}}$ by $\boldsymbol{A}$ provides:

$$\boldsymbol{L} = \boldsymbol{AU^{-1}} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & -\frac{1}{2} & 1 \end{bmatrix}$$

This gives us:

$$\boldsymbol{A} = \boldsymbol{LDL^T} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & -\frac{1}{2} & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 4 & 0 \\ 0 & 0 & 9 \end{bmatrix} \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & -\frac{1}{2} \\ 0 & 0 & 1 \end{bmatrix}$$

Where $\boldsymbol{D}$ is a diagonal matrix with the diagonal values of $\boldsymbol{U}$; the modified Cholesky factorisation of $\boldsymbol{A}$

---

d) We find a matrix $\boldsymbol{R}$ such that:

$$\boldsymbol{R} = \boldsymbol{D^{\frac{1}{2}}L^T} = \begin{bmatrix} \sqrt{1} & 0 & 0 \\ 0 & \sqrt{4} & 0 \\ 0 & 0 & \sqrt{9} \end{bmatrix} \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & -\frac{1}{2} \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 2 & -1 \\ 0 & 0 & 3 \end{bmatrix}$$

We observe that if $\boldsymbol{A\vec{x}} = \vec{\boldsymbol{b}}$ then $\boldsymbol{R^T R\vec{x}} = \vec{\boldsymbol{b}}$. Hence, if we let $\boldsymbol{R\vec{x}} = \vec{\boldsymbol{y}}$ we can solve $\boldsymbol{R^T\vec{y}} = \vec{\boldsymbol{b}}$ and then solve $\boldsymbol{R\vec{x}} = \vec{\boldsymbol{y}}$ afterwards, which is significantly easier due to $\boldsymbol{R}$ already being in upper-triangular form.

$$\boldsymbol{R^T\vec{y}} = \vec{\boldsymbol{b}} \Leftrightarrow \begin{bmatrix} 1 & 0 & 0 & : & 1 \\ 1 & 2 & 0 & : & 1 \\ 0 & -1 & 3 & : & 1 \end{bmatrix}$$

This gives us

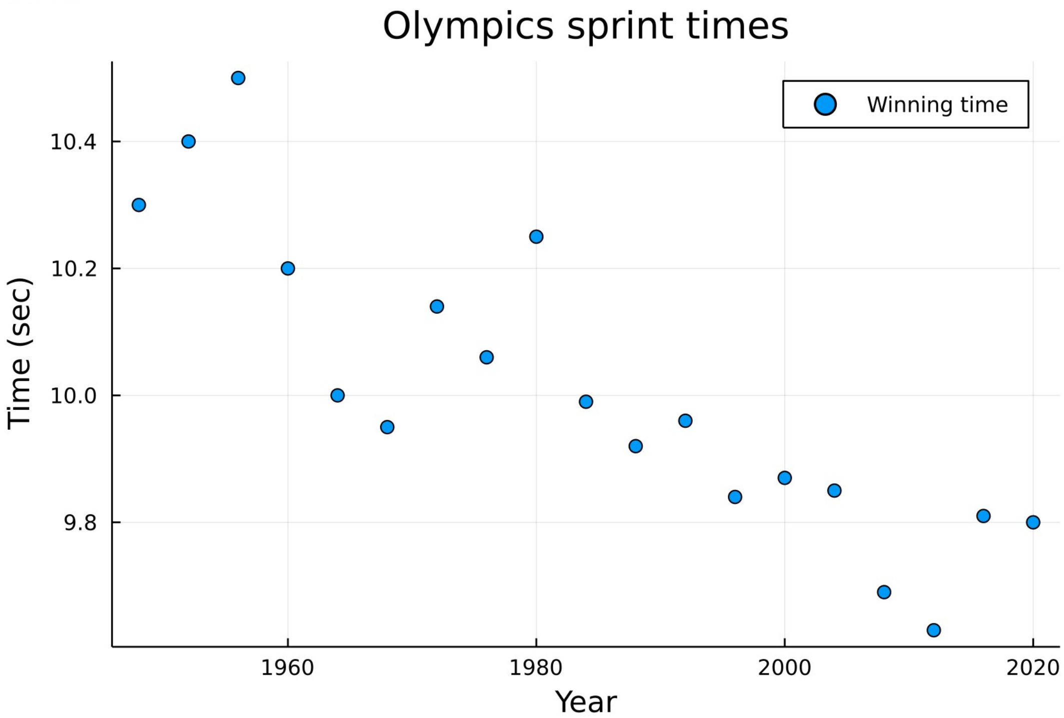
$$\vec{\boldsymbol{y}} = \begin{bmatrix} 1 \\ 0 \\ \frac{1}{3} \end{bmatrix} \quad \Rightarrow \quad \boldsymbol{R\vec{x}} = \begin{bmatrix} 1 \\ 0 \\ \frac{1}{3} \end{bmatrix}$$

$$\boldsymbol{R\vec{x}} = \vec{\boldsymbol{y}} \Leftrightarrow \begin{bmatrix} 1 & 1 & 0 & : & 1 \\ 0 & 2 & -1 & : & 0 \\ 0 & 0 & 3 & : & \frac{1}{3} \end{bmatrix} \quad \Rightarrow \quad \vec{\boldsymbol{x}} = \begin{bmatrix} \frac{17}{18} \\ \frac{1}{18} \\ \frac{1}{9} \end{bmatrix}$$

Question 5

```
results = 19×2 Matrix{Float64}:
           1948.0  10.3
           1952.0  10.4
           1956.0  10.5
           1960.0  10.2
           1964.0  10.0
           1968.0   9.95
           1972.0  10.14
              ⋮
           2000.0   9.87
           2004.0   9.85
           2008.0   9.69
           2012.0   9.63
           2016.0   9.81
           2020.0   9.8
```

```
trace1 =
```



a)

A_1 is the matrix that represents the linear approximation of T, that is $T_1 = c_0 + c_1 Y$ where Y is the vector of years, being the first column of the results matrix, and the c's are scalars (note: Y is the leftmost column of A_1). We aim to find the LSS of $A_1\vec{x} = \vec{b}$, where $\vec{b}$ is the vector of times.

```
A₁ = 19×2 Matrix{Float64}:
      1948.0  1.0
      1952.0  1.0
      1956.0  1.0
      1960.0  1.0
      1964.0  1.0
      1968.0  1.0
      1972.0  1.0
         ⋮
      2000.0  1.0
      2004.0  1.0
      2008.0  1.0
      2012.0  1.0
      2016.0  1.0
      2020.0  1.0
```

```
b =
  [10.3, 10.4, 10.5, 10.2, 10.0, 9.95, 10.14, 10.06, 10.25, 9.99, 9.92, 9.96, 9.84, 9.87, 9.8
```

We then use the qr( ) command to form the QR factorisation of A_1

```
LinearAlgebra.QRCompactWY{Float64, Matrix{Float64}}
Q factor:
19×19 LinearAlgebra.QRCompactWYQ{Float64, Matrix{Float64}}:
 -0.225239  -0.379479  -0.253539    …  -0.188642    -0.184007    -0.179371
 -0.225702  -0.337596  -0.27237         0.327949     0.370829     0.413709
 -0.226164  -0.295713   0.913028        0.0297442    0.0380811    0.0464179
 -0.226627  -0.25383   -0.0795862       0.0176489    0.0245943    0.0315397
 -0.227089  -0.211947  -0.0722007       0.00555364   0.0111075    0.0166614
 -0.227552  -0.170065  -0.0648152   …  -0.00654164  -0.00237925   0.00178315
 -0.228014  -0.128182  -0.0574297      -0.0186369   -0.015866    -0.0130951
  ⋮                                 ⋱
 -0.231252   0.164998  -0.00573123     -0.103304    -0.110273    -0.117243
 -0.231714   0.206881   0.00165426     -0.115399    -0.12376     -0.132121
 -0.232177   0.248764   0.00903975  …  -0.127495    -0.137247    -0.146999
 -0.232639   0.290647   0.0164252       0.86041     -0.150734    -0.161878
 -0.233102   0.332529   0.0238107      -0.151685     0.83578     -0.176756
 -0.233564   0.374412   0.0311962      -0.16378     -0.177707     0.808366
R factor:
2×2 Matrix{Float64}:
 -8648.58  -4.35863
     0.0   -0.0481315
```

```
Q₁, R₁ = qr(A₁)
```

$$A\vec{x} = \vec{b} \quad \Rightarrow \quad Q_1 R_1 \vec{x} = \vec{b}$$

Thus we let $R_1\vec{x} = \vec{y_1}$, and multiply both sides on the left by Q^T to get $Q_1^T\vec{b} = \vec{y_1}$

We find $Q_1^T\vec{b} = \vec{y_1}$, and then solve $R_1\vec{x} = \vec{y_1}$ by back substitution.

```
y₁ =
  [-43.6134, -1.35581, 0.223477, -0.036392, -0.196261, -0.20613, 0.0240012, -0.0158677, 0.21
```

```
y₁ = Q₁'b
```

```
c0₁ = 28.16898245614025
```

```
c0₁=y₁[2]/R₁[2,2]
```

```
c1₁ = -0.009153508771929773
```

```
c1₁=(y₁[1]-R₁[1,2]*c0₁)/R₁[1,1]
```

```
x₁ =   [-0.00915351, 28.169]
```

```
x₁=[c1₁; c0₁]
```

Using the backslash command $\vec{x} = A_1\backslash\vec{b}$ yields the same result:

```
[-0.00915351, 28.169]
```

```
A₁\b
```

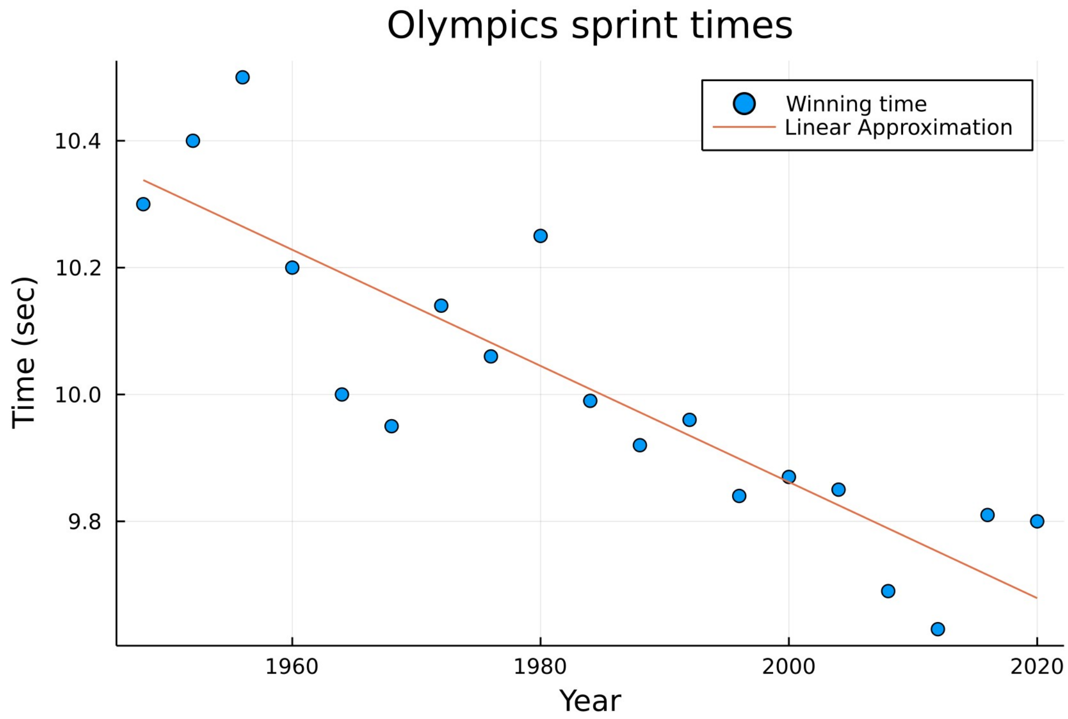
In both cases, this yields a function

$$t = 28.169 - 0.00915351y$$

to predict a time, t, in a specific year, y.

T_1 represents the outputs of this function, that is the predicted results from the degree one fitting polynomial.

```
T₁ =
 [10.3379, 10.3013, 10.2647, 10.2281, 10.1915, 10.1549, 10.1183, 10.0816, 10.045, 10.0084,
```



The error of the liner approximation was calculated by the 2-norm of $\vec{b} - T_1$

```
error₁ = 0.49489162379075124

error₁=norm((b-T₁),2)
```

---

b)

A_2 is the matrix that represents the quadratic approximation of T, that is $T_2 = c_0 + c_1 Y + c_2 Y^2$ where Y is the vector of years, and the c's are scalars (note: Y^2 is the leftmost column of A_2). We aim to find the LSS of $A_2\vec{x} = \vec{b}$, where $\vec{b}$ is the vector of times.

```
A₂ = 19×3 Matrix{Float64}:
      3.7947e6    1948.0  1.0
      3.8103e6    1952.0  1.0
      3.82594e6   1956.0  1.0
      3.8416e6    1960.0  1.0
      3.8573e6    1964.0  1.0
      3.87302e6   1968.0  1.0
      3.88878e6   1972.0  1.0
      ⋮
      4.0e6       2000.0  1.0
      4.01602e6   2004.0  1.0
      4.03206e6   2008.0  1.0
      4.04814e6   2012.0  1.0
      4.06426e6   2016.0  1.0
      4.0804e6    2020.0  1.0
```

We then use the qr( ) command to form the QR factorisation of A_2

```
LinearAlgebra.QRCompactWY{Float64, Matrix{Float64}}
Q factor:
19×19 LinearAlgebra.QRCompactWYQ{Float64, Matrix{Float64}}:
 -0.221085  -0.377616  -0.441581   -0.236039    …  -0.216224   -0.236503   -0.259898
 -0.221994  -0.337179  -0.29522    -0.230205        0.324933    0.366319    0.407516
 -0.222904  -0.296572  -0.166029   -0.018153       -0.110514   -0.241514   -0.390213
 -0.223817  -0.255797  -0.0540085   0.925913        0.0215711   0.032055    0.0429853
 -0.224732  -0.214853   0.0408413  -0.0731088       0.0210634   0.0414732   0.0637639
 -0.225648  -0.17374    0.118521   -0.0711444   …   0.0182656   0.0463712   0.0775093
 -0.226566  -0.132458   0.179029   -0.0681935       0.0131779   0.0467488   0.0842214
  ⋮                                             ⋱
 -0.233046   0.161245   0.121817   -0.0199155      -0.0865566  -0.0771763  -0.0657236
 -0.233979   0.203879   0.0449614  -0.00907265     -0.109964   -0.112961   -0.115277
 -0.234914   0.246681  -0.0490644   0.00275668  …  -0.135662   -0.153266   -0.171864
 -0.235851   0.289652  -0.160261    0.0155725       0.83635    -0.198092   -0.235484
 -0.236789   0.332792  -0.288627    0.0293748      -0.193928    0.752563   -0.306137
 -0.23773    0.376101  -0.434165    0.0441636      -0.226496   -0.301303    0.616177
R factor:
3×3 Matrix{Float64}:
 -1.7164e7  -8648.06    -4.35784
  0.0         -95.4568  -0.0962336
  0.0           0.0     -0.000473472
```

```
Q₂, R₂ = qr(A₂)
```

Using the same logic as in 5a) we find $Q_2^T\vec{b} = \vec{y_2}$, and then solve $R_2\vec{x} = \vec{y_2}$

```
y₂ =
  [-43.5958, -1.83598, -0.121426, -0.0179809, -0.150011, -0.137473, 0.109632, 0.0813037, 0.3
```

```
y₂=Q₂'b
```

```
c0₂ = 256.4576261224542
```

```
c0₂ = y₂[3] / R₂[3,3]
```

```
c1₂ = -0.2393112560796893
```

```
c1₂ = (y₂[2] - R₂[2,3]*c0₂) / R₂[2,2]
```

```
c2₂ = 5.8003464543286806e-5
```

```
c2₂ = (y₂[1] - R₂[1,3]*c0₂ - R₂[1,2]*c1₂) / R₂[1,1]
```

```
x₂ =  [5.80035e-5, -0.239311, 256.458]
```

```
x₂=[c2₂, c1₂, c0₂]
```

Using the backslash command $\vec{x} = A_2\backslash\vec{b}$ yields the same result:

```
[5.80035e-5, -0.239311, 256.458]
```

```
A₂\b
```

In both cases, this yields a function

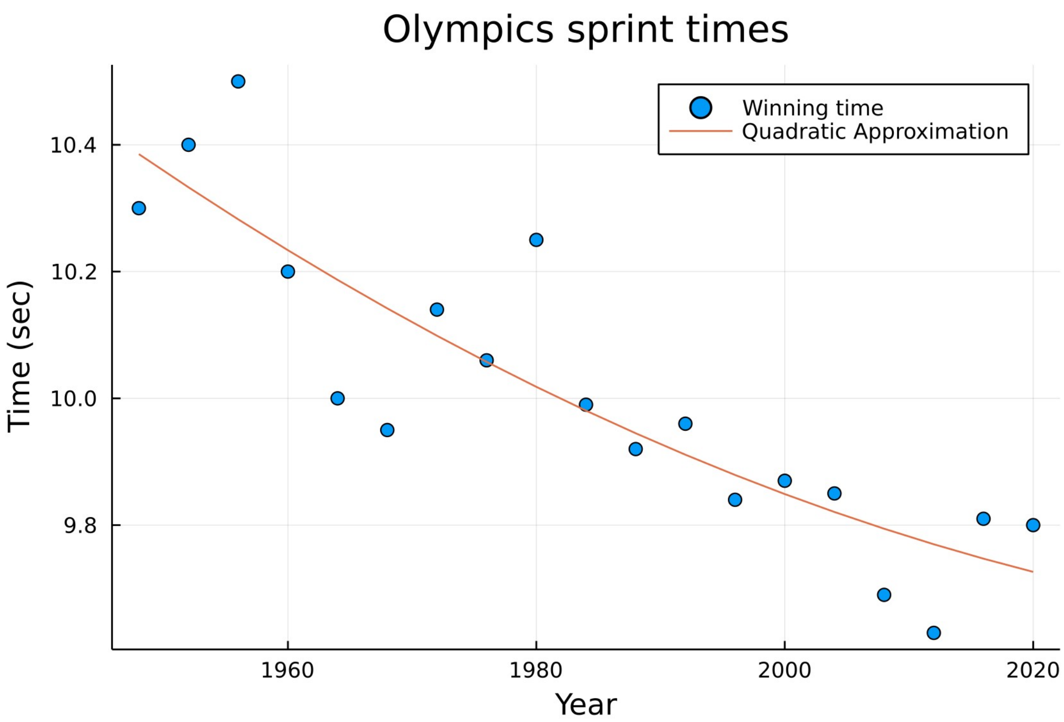
$$t = 256.458 - 0.239311y + 5.80035 * 10^{-5}y^2$$

to predict a time, t, in a specific year, y.

T_2 represents the outputs of this function, that is the predicted results from the degree two fitting polynomial.

```
T₂ =
  [10.3853, 10.3329, 10.2824, 10.2337, 10.1869, 10.1419, 10.0988, 10.0575, 10.0181, 9.98058,
```

```
T₂= A₂*x₂
```



The error of the quadratic approximation was calculated by the 2-norm of $\vec{b} - T_2$

```
error₂ = 0.4829425446164058
```

```
error₂=norm((b-T₂),2)
```

---

c)

A_3 is the matrix that represents the linear approximation of T, that is $T_3 = c_0 + c_1 Y + c_2 Y^2 + c_3 Y^3$ where Y is the vector of years, and the c's are scalars (note: Y^3 is the leftmost column of A_3). We aim to find the LSS of $A_3\vec{x} = \vec{b}$, where $\vec{b}$ is the vector of times.

```
A₃ = 19×4 Matrix{Float64}:
      7.39208e9  3.7947e6   1948.0  1.0
      7.43771e9  3.8103e6   1952.0  1.0
      7.48353e9  3.82594e6  1956.0  1.0
      7.52954e9  3.8416e6   1960.0  1.0
      7.57573e9  3.8573e6   1964.0  1.0
      7.62211e9  3.87302e6  1968.0  1.0
      7.66868e9  3.88878e6  1972.0  1.0
      ⋮
      8.0e9      4.0e6      2000.0  1.0
      8.0481e9   4.01602e6  2004.0  1.0
      8.09638e9  4.03206e6  2008.0  1.0
      8.14487e9  4.04814e6  2012.0  1.0
      8.19354e9  4.06426e6  2016.0  1.0
      8.24241e9  4.0804e6   2020.0  1.0
```

```
A₃=hcat(results[:,1].^3, A₂)
```

We then use the qr( ) command to form the QR factorisation of A_3

```
LinearAlgebra.QRCompactWY{Float64, Matrix{Float64}}
Q factor:
19×19 LinearAlgebra.QRCompactWYQ{Float64, Matrix{Float64}}:
 -0.216954  -0.375711  -0.441016   …  -0.225673   -0.149309    -0.0320494
 -0.218293  -0.3367    -0.297096       0.320596    0.417887     0.539945
 -0.219638  -0.29736   -0.169512      -0.105651   -0.243414    -0.402101
 -0.220988  -0.257688  -0.0583686      0.0334019  -0.116505    -0.33859
 -0.222344  -0.217685   0.036231       0.0227671   0.0160776   -0.000886163
 -0.223705  -0.17735    0.114183   …   0.0200156   0.0188647    0.00761503
 -0.225072  -0.13668    0.175383       0.0145947   0.0228684    0.0236923
  ⋮                                ⋱
 -0.234796   0.157433   0.126009      -0.0895215  -0.0378731    0.0349074
 -0.236208   0.200809   0.0494556     -0.112624   -0.0767789   -0.0227446
 -0.237625   0.244527  -0.0447842  …  -0.137477   -0.127629    -0.106405
 -0.239048   0.28859   -0.156814       0.83602    -0.191885    -0.219805
 -0.240477   0.332997  -0.286739      -0.192029    0.728991    -0.366677
 -0.241911   0.37775   -0.434661      -0.221524   -0.366464     0.449248
R factor:
4×4 Matrix{Float64}:
 -3.40721e10  -1.7163e7    -8646.48      -4.35651
  0.0         -1.89305e5   -190.827      -0.144282
  0.0          0.0           -0.939094   -0.00142016
  0.0          0.0            0.0         4.54134e-6
```

```
Q₃, R₃ = qr(A₃)
```

Using the same logic as in 5a) and 5b) we find $Q_3^T\vec{b} = \vec{y_3}$, and then solve $R_3\vec{x} = \vec{y_3}$

```
y₃ =
 [-43.5729, -2.3154, -0.139728, -0.0251058, -0.153368, -0.14093, 0.106432, 0.0786361, 0.315
```

```
c0₃ = -5528.273024402207
```

```
c0₃ = y₃[4] / R₃[4,4]
```

```
c1₃ = 8.509036814593857
```

```
c1₃= (y₃[3] - R₃[3,4]*c0₃) / R₃[3,3]
```

```
c2₃ = -0.0043517676180647846
```

```
c2₃ = (y₃[2] - R₃[2,4]*c0₃ - R₃[2,3]*c1₃) / R₃[2,2]
```

```
c3₃ = 7.408889587719471e-7
```

```
c3₃ = (y₃[1] - R₃[1,4]*c0₃ - R₃[1,3]*c1₃ - R₃[1,2]*c2₃) / R₃[1,1]
```

```
x₃ =  [7.40889e-7, -0.00435177, 8.50904, -5528.27]
```

```
x₃ = [c3₃; c2₃; c1₃; c0₃]
```

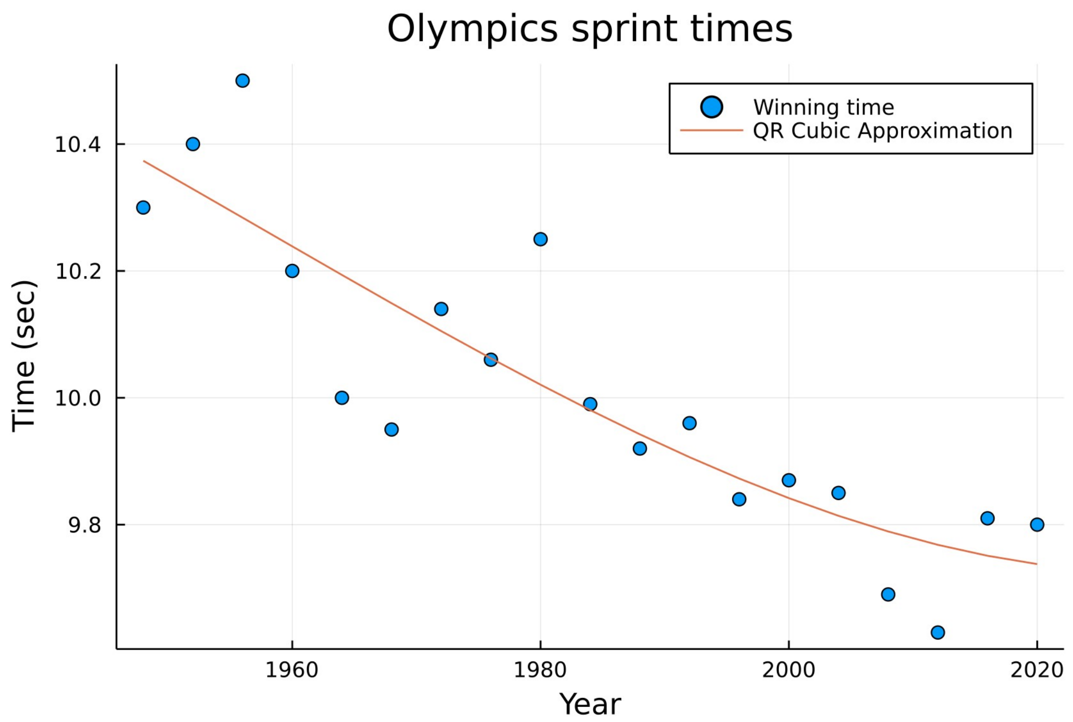
The QR factorisation yields a function

$$t = -5528.27 + 8.50904y - 0.00435177y^2 + 7.40889 * 10^{-7}y^3$$

to predict a time, t, in a specific year, y.

T_3 represents the outputs of this function, that is the predicted results from the degree three fitting polynomial with the QR factorization.

```
T₃ =
 [10.3737, 10.329, 10.2839, 10.2387, 10.1937, 10.1491, 10.1051, 10.0622, 10.0206, 9.98058,
```



The error of this cubic approximation was calculated by the 2-norm of $\vec{b} - T_3$

```
error₃ = 0.4822274371091855
• error₃ = norm((b-T₃),2)
```

In this case, using the backslash command yielded a different LSS. This is because we have violated the assumption of the QR factorized matrix having full rank (part 4.1 in the lecture notes), with A_3 having rank 3 and thus a nullity of 1. As a result, the QR factorization does not occur correctly, providing an incorrect LSS.

The rank of A_3 is 3, as seen by the Julia rank command.

```
r = 3
• r = rank(A₃)
```

```
x3 = [3.29113e-8, -0.000137756, 0.14879, 0.000225011]
• x3 = A₃\b
```

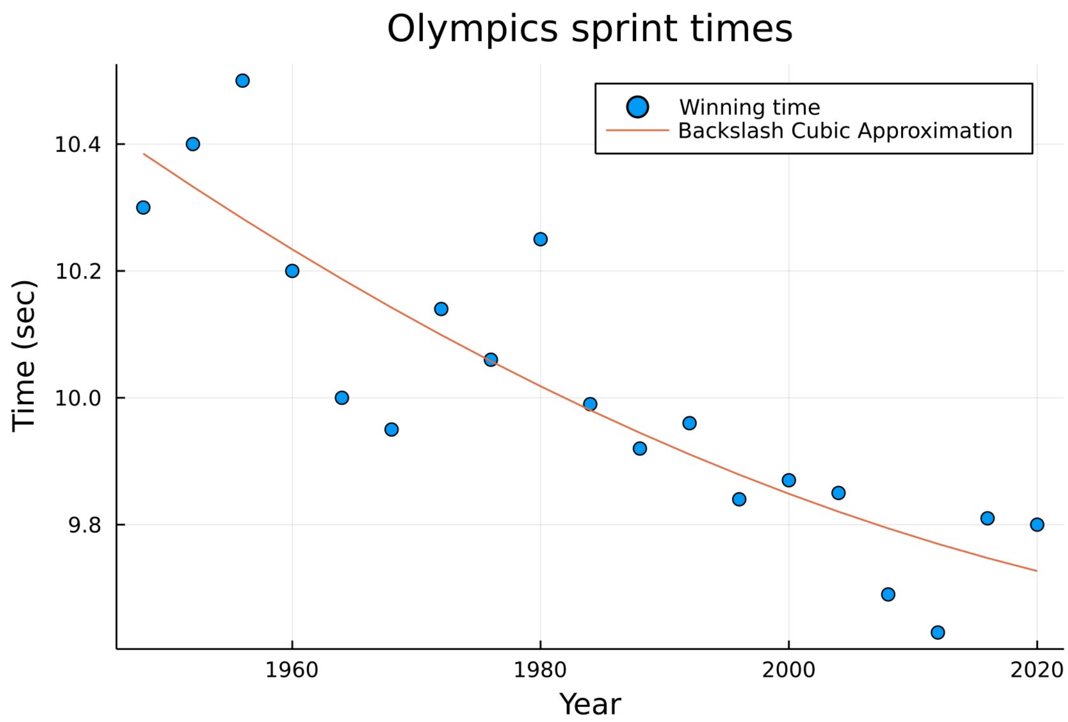
The backslash command yields a function

$$t = 0.000225011 + 0.14879y - 0.000137756y^2 + 3.291131 \times 10^{-8}y^3$$

to predict a time, t, in a specific year, y

$T3$, not to be confused with T_3 represents the outputs of this function, that is the predicted results from the degree three fitting polynomial using the backslash command.

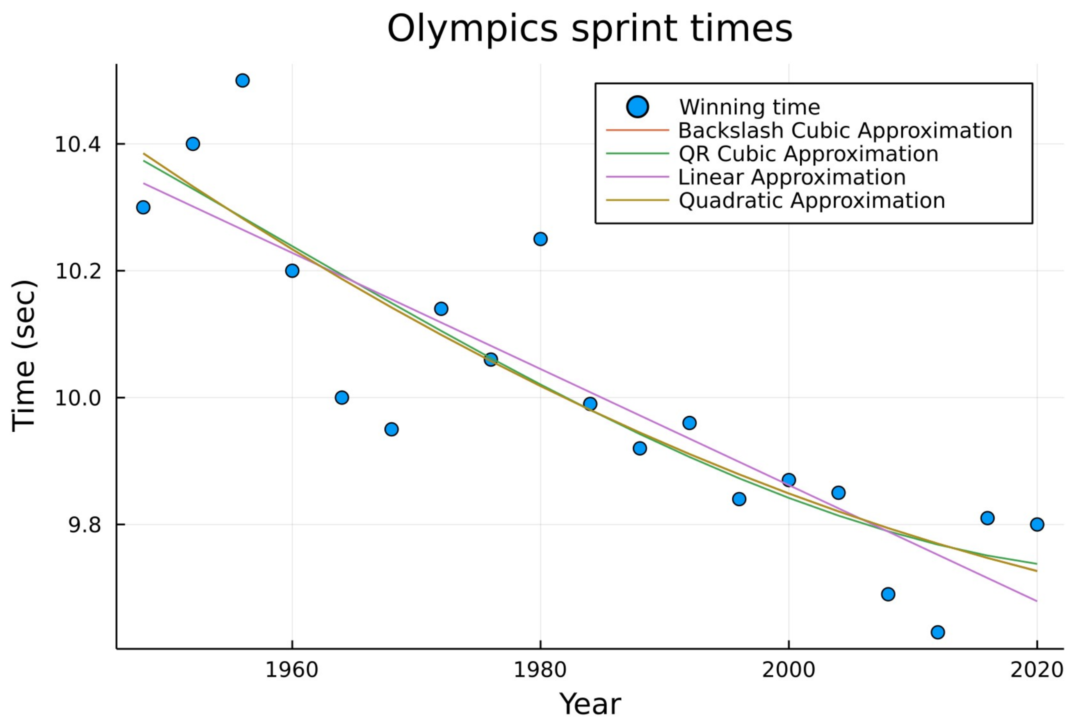
```
T3 =
 [10.3849, 10.3328, 10.2825, 10.2339, 10.1871, 10.1422, 10.099, 10.0577, 10.0182, 9.98052,
• T3 = A₃*x3
```



The error of this cubic approximation was calculated by the 2-norm of $\vec{b} - T3$

```
error3 = 0.48288052556399447
error3 = norm((T3-b),2)
```

d)



Note that the x^3 coefficient of both our cubic models is virtually zero. Because of this, the cubic approximations are virtually the same as the quadratic approximation. From this we can conclude there is very little benefit to actually adding a cubic approximation.

e) 2024 predictions are simply found by setting x to 2024 in the specific functions, which is the same as assessing the dot product of the vector

$$\vec{Y_i} = [2024^i \quad 2024^{i-1} \quad \dots \quad 2024^1 \quad 1]^T$$

and the specific solution vector $\vec{x_i}$ for each degree i approximation, up to degree three. The difference between the quadratic and cubic approximations, especially between the quadratic and backslash cubic approximations, demonstrates how the approximations give the same result to a hundredth, or even less, of a second, further highlighting the similarities we see in their graphic representations.

Linear approximation in seconds:

$[9.64228]$

Quadratic approximation in seconds:

$[9.70724]$

Cubic approximation in seconds, using QR:

$[9.72915]$

Cubic approximation in seconds, using the backslash command:

$[9.70836]$