Nolan Dahlman - Project 10.
Numerical methods applied to Real World Data

Project Goal

I will combine all the information I have learned about interpolating polynomials to explain what it means to approximate this data, how it can be helpful, and begin a deep dive into the meaning behind the data.

Introduction

My data is temperature variation from the average global temperature in each year. My focus will be to approximate the next few years of data from my interpolation methods. I will use Taylor and Lagrange interpolation methods to predict the future changes in global temperature to see if we are trending to a hotter or cooler global climate.

Analysis

This data is a investigation on temperature anomalies in the global temperatures for earth. The anomaly is the highest or lowest temperature as compared to a baseline temperature which is averaged over twenty or thirty years depending on the data set that you are looking at. The data I will be looking at is averaged over a century. This data is important because If you look at data at a specific station you could have different temperatures for areas with higher or lower elevation. Using anomalies will help make those factors less influential on the data.

The interval that we will be looking at this data on will be yearly. I am starting my data set with years since 1985 with the hope that I can add more data to the set, when I can closely approximate the future years for more accurate predictions. Yearly is how the global temperature anomalies are measured at the world temperature stations.

The goal I have in this exploring this data is to gain my own understanding of this field. A lot of people use this science in daily conversation and exploring this data set will not only help me come to my own conclusion, but it will also help me understand terms and other data and the influences of this data. Overall it will help me feel as though I am educated on the subject.

Interpolation Methods

The first topic in this conversation will be interpolation methods. I have learned many different methods to interpolate data with polynomials. Each one has it's own positives and negatives. We use interpolation methods to fill in the gaps of the data. For our data we are looking at yearly average global surface temperature anomalies. So an approximation between two points would mean what the average surface temperature anomaly was doing between years.

The first method we will talk about is the Taylor approximation. The Taylor approximation was the first method we used to approximate data. Although this Interpolation method seems trivial in comparison to other methods we have learned since, it is an important method to understand. Taylor's Theorem actually helps us with our Hermite interpolation. The order of the Taylor polynomial is low and only travels through one point. This does not help us far from that singular point we used to create the interpolation. This is a good starting point for interpolation but it does not accurately represent the global average temperature anomaly clearly thus we must disregard this method for any predictive behavior.

The next interpolation method we use is the Lagrange Interpolation method. The Lagrange interpolation method expands on the ideas introduced by the Taylor interpolation. However, the Lagrange interpolation uses every data point in it's interpolation method. Thus, the Lagrange does a better job of interpolating each data point unlike the Taylor. This is advantageous because we get a polynomial that approximates each anomaly point exactly. When we use the Lagrange interpolation we learn something new about interpolating polynomials that we didn't know before. When we have many data points our polynomial begins to incorporate large oscillations on either end of the interval of known data from the polynomial's high degree. We can know discuss the pros and cons of this method. Although Lagrange interpolates the anomaly points well, we see that in-between our known data is very poor approximation of what is happening. Further, Because these oscillations happen at the end of the known data interval we have to also disregard this as an approximation method for future values.

We then moved on to the Hermite interpolation method which incorporates the derivative through a method of numerical differentiation. However, involving the derivative does something similar to the Lagrange and begins to show oscillations even more extremely. Prediction here also do not have much confidence behind them.

Lastly, we look at our number one candidate for interpolating our data, the cubic Spline. The Cubic Spline is different from the other interpolation methods because it uses a piece-wise function from point to point that each element has a very low comparative degree. Thus, the approximations between points are very good. This method is the first method we see this. The disadvantages of this interpolation method is that it has no ability to predict future values because it requires two points to make a piece of interpolations.

So, what have we learned from interpolation methods? The first thing we can say about interpolation is that more data is not always what we want. We can see that our interpolation methods don't model our data well when we use all of our data. The question comes about, what happens if we use less data points and how can we optimize this process for the best approximation of the global temperature anomalies inside and outside the interval?

Numerical Differentiation

Numerical Differentiation comes from our Taylor's Theorem. The importance of this technique is that we do not need a function to take a derivative. We can use anomaly points around the data to approximate the derivative. Generally, the more points we use around our point the better approximation we can get. When we do this with our data we see that the derivatives at each point have a net increase over the interval. What does this mean for our data? The derivative of the global temperature anomalies is the rate of change of the temperature anomaly from year to to year. An increase would imply hotter anomalies in the future and negative would mean vice versa. From this technique we learn that in our data we have signs pointing to hotter temperature anomalies which does not bode well for the surface temperature of the Earth.

Numerical Integration

Unfortunately, this numerical method doesn't help us with our data much because we are comparing temperature anomalies to a global average. If the data we were looking at was just the global average then it would make sense to look at the integral. However, One of these techniques that is applied to only the integral of a function and not data can help us with our interpolation methods. This method would be the Gaussian Quadrature Method.

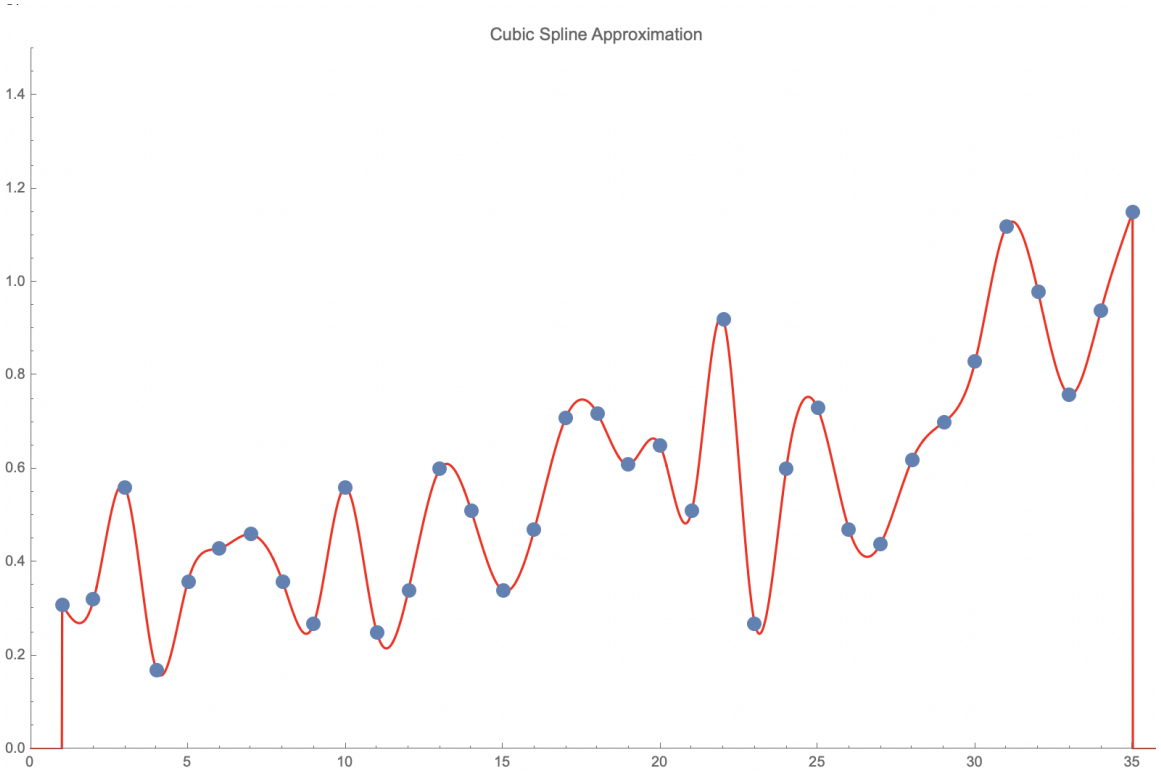
Without going into so much detail the Gaussian Quadrature Method is a method of integration that uses specific points and weights to approximate an integral of a low order polynomial exactly. What we learn

from the Gaussian quadrature method is where these specific points are placed. The density of these points are high on the edges of our known data interval and spars in the center of our known interval data. Maybe we can use this idea from the Gaussian Quadrature method to pick points that will help our interpolation method have less oscillations with less, better placed points.

Results

As compared to our analysis we see that we get what we expected lets first look at our interpolation methods. We begin by looking at our Cubic Spline interpolation.

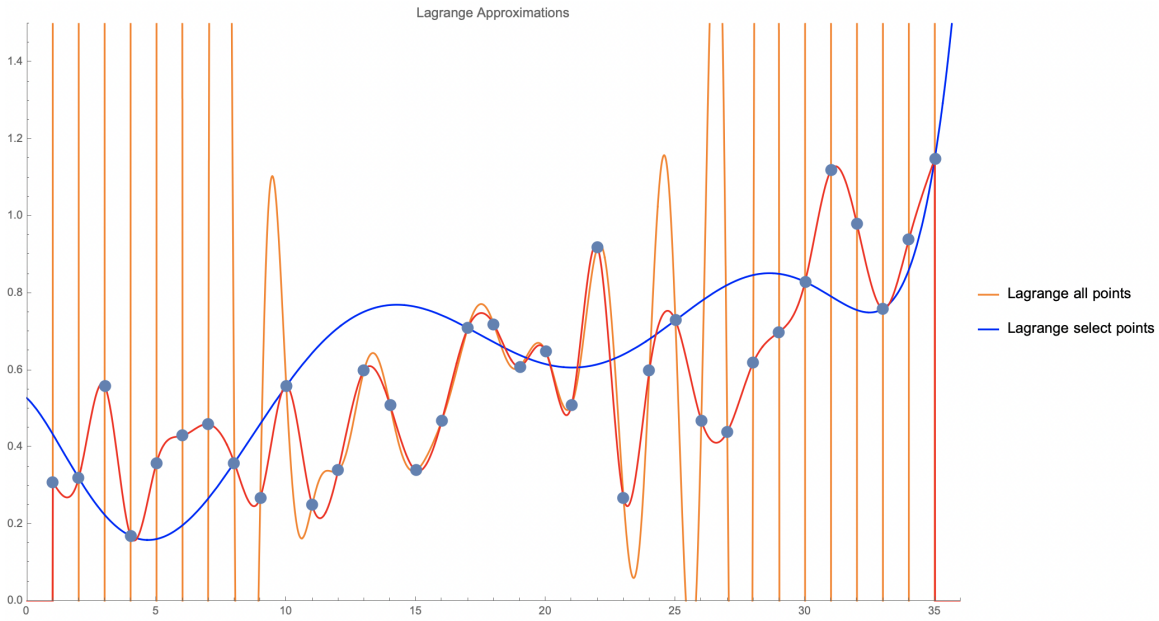
Cubic Spline



Our Cubic Spline approximation, like we thought, approximates the data well on the given interval. We noted that this is a good approximation for the given data, but nothing further than the given data. We will use this great approximation on the interval to compare the two other methods that we have to it.

Lagrange

We then look at our Lagrange we notice that we do have too many points and we do get these oscillations that we talked about. Since we knew this was going to happen we also used our ideas from numerical integration to pick points for our Lagrange that hopefully approximate the future better.



When we started this project it was 2021, we now have data for 2022 and we can use the 2022 data to see which of these two interpolations predict the "future" point. When we used all the points the Lagrange estimated the 36^{th} point to be -1.71802×10^9 which is nowhere near what we would expect the next anomaly to be. When we used less, more specific points we note that the estimated 2022 point was 1.75192. The actual data point was 1.45.

From these two interpolations we see that Lagrange that estimates all the points does a good job of estimating all the points on the interval but not in-between those points and also not a good prediction. The Lagrange that does not use all the points does not estimate all the points well, but does better at estimating in-between the points better. It also estimates the future point better that the all the points Lagrange.

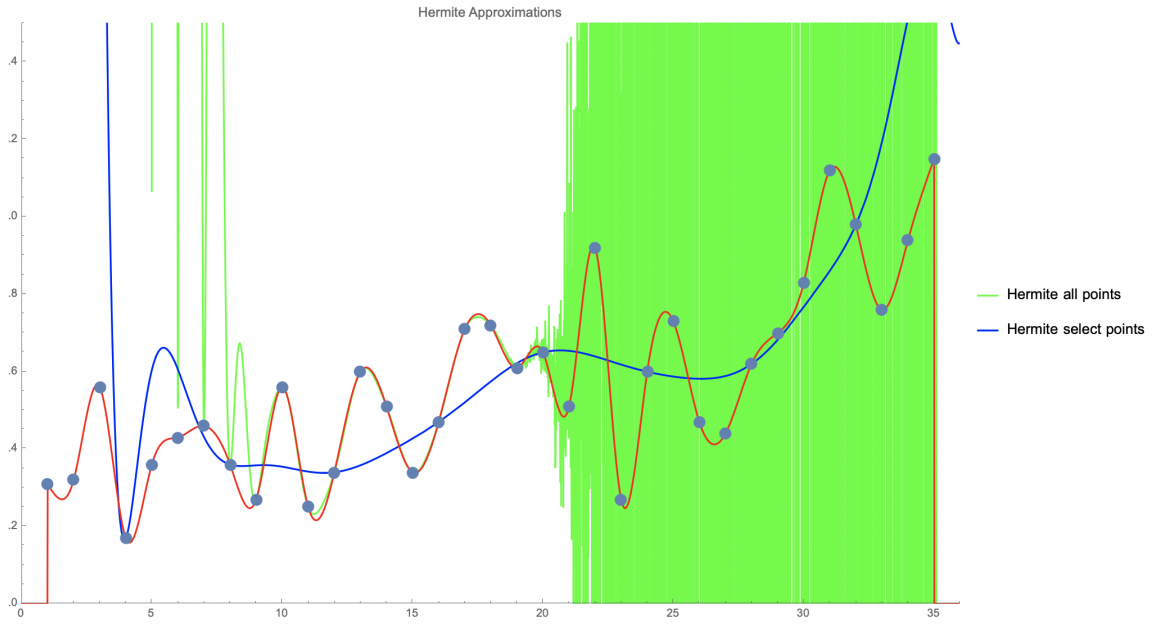
Lagrange Error

Since our Spline approximates the data best on the interval, we will use it to look at the an expected error between the data points

Looking at the spaces in-between points we see that the Lagrange that uses less points stays within an acceptable error. Compared to the Lagrange with every point which oscillates at an extreme rate would have a bad error. Although the all points Lagrange estimates the points better than the select points Lagrange we note that the space in-between points is unforgivable.

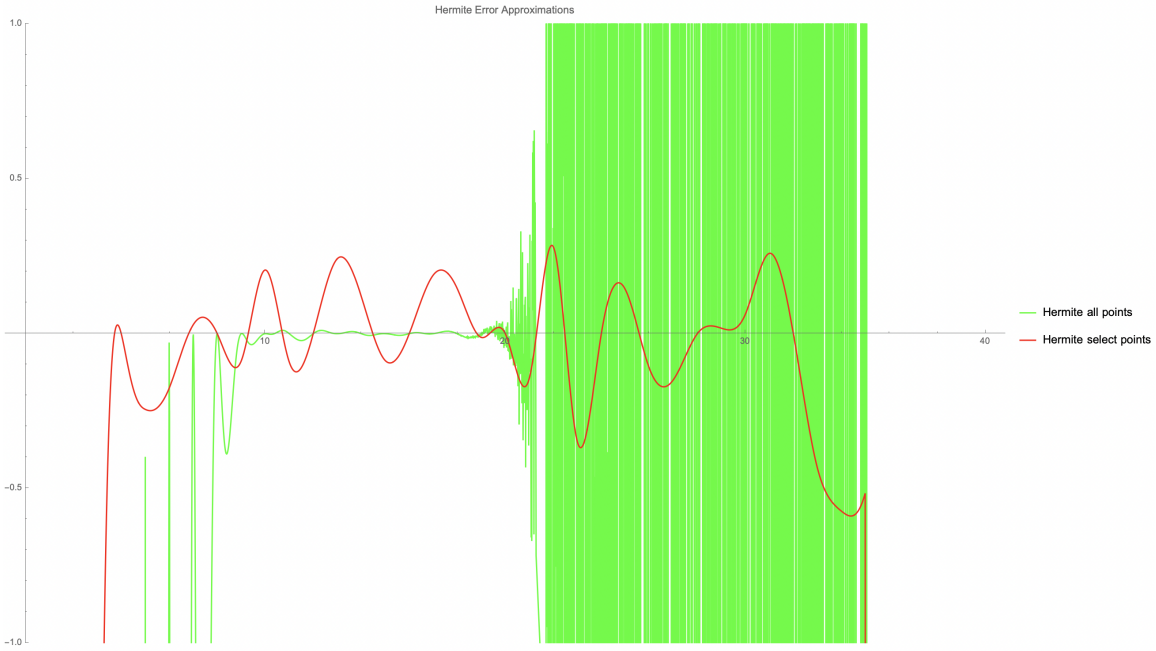
Hermite

When Looking at the Hermite Interpolations we know that we will have a similar problem to the Lagrange but to a higher degree. We look at the graph.



We notice from this graph that the Hermite from with every data point is clearly unusable. The amount of extreme oscillations is unbearably wrong and should not be considered an interpolation we can rely on. On the other hand the more manageable Hermite that uses less points is clearly an better approximation but at the end points we can still see these oscillations and therefore should not be considered as a viable option for predictions.

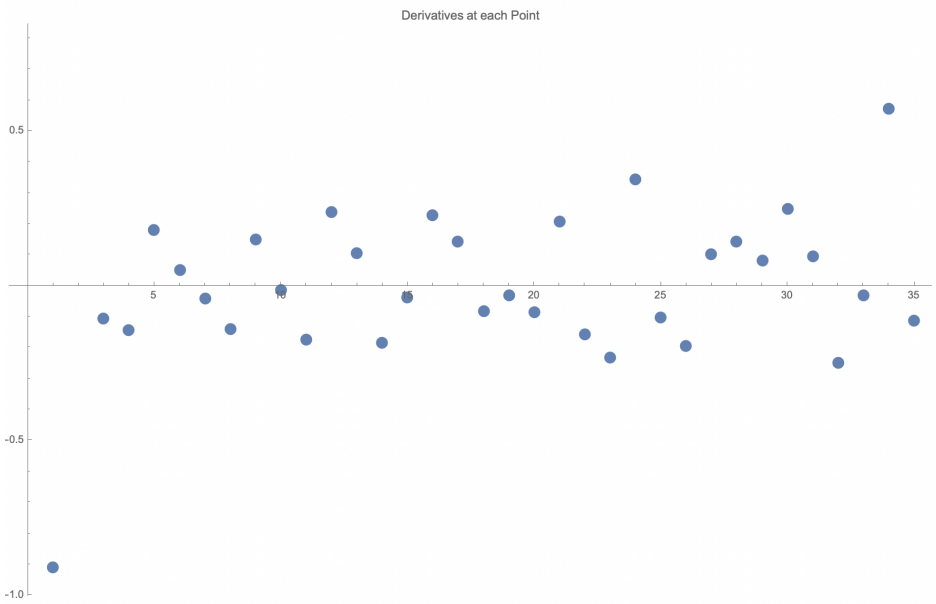
Hermite Error



From the graph we can see that the interpolation with less points approximates the points well. The error that we look at is the spaces in-between the points. It is obvious that the select points Hermite has less error on the space in-between points that the all points Hermite. Noting this we can say that the select point Hermite's error is much better than that of the all points Hermite.

Numerical Differentiation

Numerical Differentiation might be the most valuable insight for us in this data set. The reason why is because if we can see a trend in the derivatives of the temperature anomaly we can say that this would be a trend that we can expect to continue in the future. So we find the derivatives of the data and graph them.



When Looking at the derivatives we see that they are all over the place. Some derivatives are negative (getting colder) and some are positive (getting warmer), but the real test of this data will be to sum all the derivatives to see which of them prevails. When we do this we are left with a positive derivative sum. Thus

from this we see that over the last thirty-five year we have had more increases in temperature anomalies than decreases. This is a trend that we can pull from the derivative data.

Conclusion

To be honest, if you got this far I'm really tired of doing this assignment so just tell me I could work on my conclusion and that I need to add a prediction. And a Less wordy analysis section so that it is actually readable.

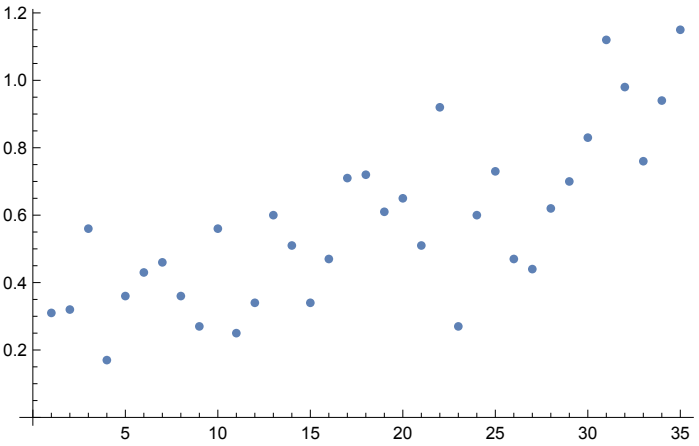
Project 10

In[654]:=
```
xi = Table[{}, 2021 - 1986];
For[i = 1, i ≤ 35, i++,
  xi[[i]] = i
 ];
xi
fxi = {0.31, 0.32, 0.56, 0.17, 0.36, 0.43, 0.46, 0.36, 0.27, 0.56, 0.25,
   0.34, 0.60, 0.51, 0.34, 0.47, 0.71, 0.72, 0.61, 0.65, 0.51, 0.92, 0.27,
   0.60, 0.73, 0.47, 0.44, 0.62, 0.70, 0.83, 1.12, 0.98, 0.76, 0.94, 1.15};
(*Just a check to see if we have the same amount of data points
 and defining a vcariable for the legnth of the vectors*)
n = Length[xi];
Length[fxi];
Orderedpairs = Table[{}, 2021 - 1986];
For[i = 1, i ≤ n, i++,
  Orderedpairs[[i]] = {xi[[i]], fxi[[i]]}
 ];
Orderedpairs
(*Graphing the Points*)
ListPlot[Orderedpairs]
```

Out[656]=
```
{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18,
 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35}
```

Out[662]=
```
{{1, 0.31}, {2, 0.32}, {3, 0.56}, {4, 0.17}, {5, 0.36}, {6, 0.43}, {7, 0.46},
 {8, 0.36}, {9, 0.27}, {10, 0.56}, {11, 0.25}, {12, 0.34}, {13, 0.6}, {14, 0.51},
 {15, 0.34}, {16, 0.47}, {17, 0.71}, {18, 0.72}, {19, 0.61}, {20, 0.65}, {21, 0.51},
 {22, 0.92}, {23, 0.27}, {24, 0.6}, {25, 0.73}, {26, 0.47}, {27, 0.44}, {28, 0.62},
 {29, 0.7}, {30, 0.83}, {31, 1.12}, {32, 0.98}, {33, 0.76}, {34, 0.94}, {35, 1.15}}
```

Out[663]=



Numerical Differentiation with the Taylor Theorem

In[664]:=
```
f[x_] := xi[[1]] + derivfxi[[1]] ((x - xi[[1]]) / 2)
derivfxi = Table[{}, n];
Sderivfxi = Table[{}, n];
h = xi[[2]] - xi[[1]];

For[i = 3, i ≤ n - 2, i++,
 derivfxi[[i]] =
  (fxi[[i - 2]] - 8 fxi[[i - 1]] +
     8 fxi[[i + 1]] - fxi[[i + 2]]) / (12 h)]

For[i = 1, i ≤ 2, i++,
 derivfxi[[i]] =
  (-25 fxi[[i]] + 48 fxi[[i + 1]] - 36 fxi[[i + 2]] +
     16 fxi[[i + 3]] - 3 fxi[[i + 4]]) (1 / (12 h))]
For[i = n, i ≥ n - 1, i--,
 derivfxi[[i]] =
  (-25 fxi[[i]] + 48 fxi[[i - 1]] - 36 fxi[[i - 2]] +
     16 fxi[[i - 3]] - 3 fxi[[i - 4]]) (1 / (12 (-h)))]

derivfxi

(*For loops to tanspose the derivative values into the Table*)
For[i = 3, i ≤ n - 2, i++,
 Sderivfxi[[i]] =
  (derivfxi[[i - 2]] - 8 derivfxi[[i - 1]] +
     8 derivfxi[[i + 1]] - derivfxi[[i + 2]]) / (12 h)]
For[i = 1, i ≤ 2, i++,
 Sderivfxi[[i]] =
  (-25 derivfxi[[i]] + 48 derivfxi[[i + 1]] - 36 derivfxi[[i + 2]] +
     16 derivfxi[[i + 3]] - 3 derivfxi[[i + 4]]) (1 / (12 h))]
For[i = n, i ≥ n - 1, i--,
 Sderivfxi[[i]] =
  (-25 derivfxi[[i]] + 48 derivfxi[[i - 1]] - 36 derivfxi[[i - 2]] +
     16 derivfxi[[i - 3]] - 3 derivfxi[[i - 4]]) (1 / (12 (-h)))]
Sderivfxi
```

Out[671]= {-0.909167, 1.43583, -0.104167, -0.1425, 0.181667, 0.0508333, -0.0391667, -0.1375, 0.150833, -0.0116667, -0.174167, 0.2375, 0.105833, -0.184167, -0.0358333, 0.229167, 0.144167, -0.0816667, -0.03, -0.0833333, 0.208333, -0.155833, -0.231667, 0.344167, -0.100833, -0.195, 0.1025, 0.143333, 0.0833333, 0.25, 0.095, -0.249167, -0.0291667, 0.5725, -0.110833}

Out[675]= {7.71451, -2.75097, -1.14313, 0.305972, 0.123472, -0.147639, -0.122986, 0.131875, 0.0951389, -0.247917, 0.169861, 0.201042, -0.292639, -0.09375, 0.272361, 0.111458, -0.207708, -0.0900694, -0.00645833, 0.165069, -0.0315278, -0.328958, 0.359097, 0.0904861, -0.387292, 0.152292, 0.210208, -0.0498611, 0.0717361, 0.0404861, -0.323403, -0.109653, 0.564931, 0.497708, -2.25243}

Making Taylor Polynomial

In[676]:=
```
Allptay = Table[ {}, n] ;
For[ i = 1, i ≤ n, i++,
 Allptay[[i]] = fxi[[i]] +
   derivfxi[[i]] ((x - xi[[i]]) / i !) + Sderivfxi[[i]] ((x - xi[[i]]) ^ (2) / i !)
]
Simplify[Allptay];
Taylorpol[x_] = Simplify[Allptay[[1]]];
Taylorpol[xi]
```

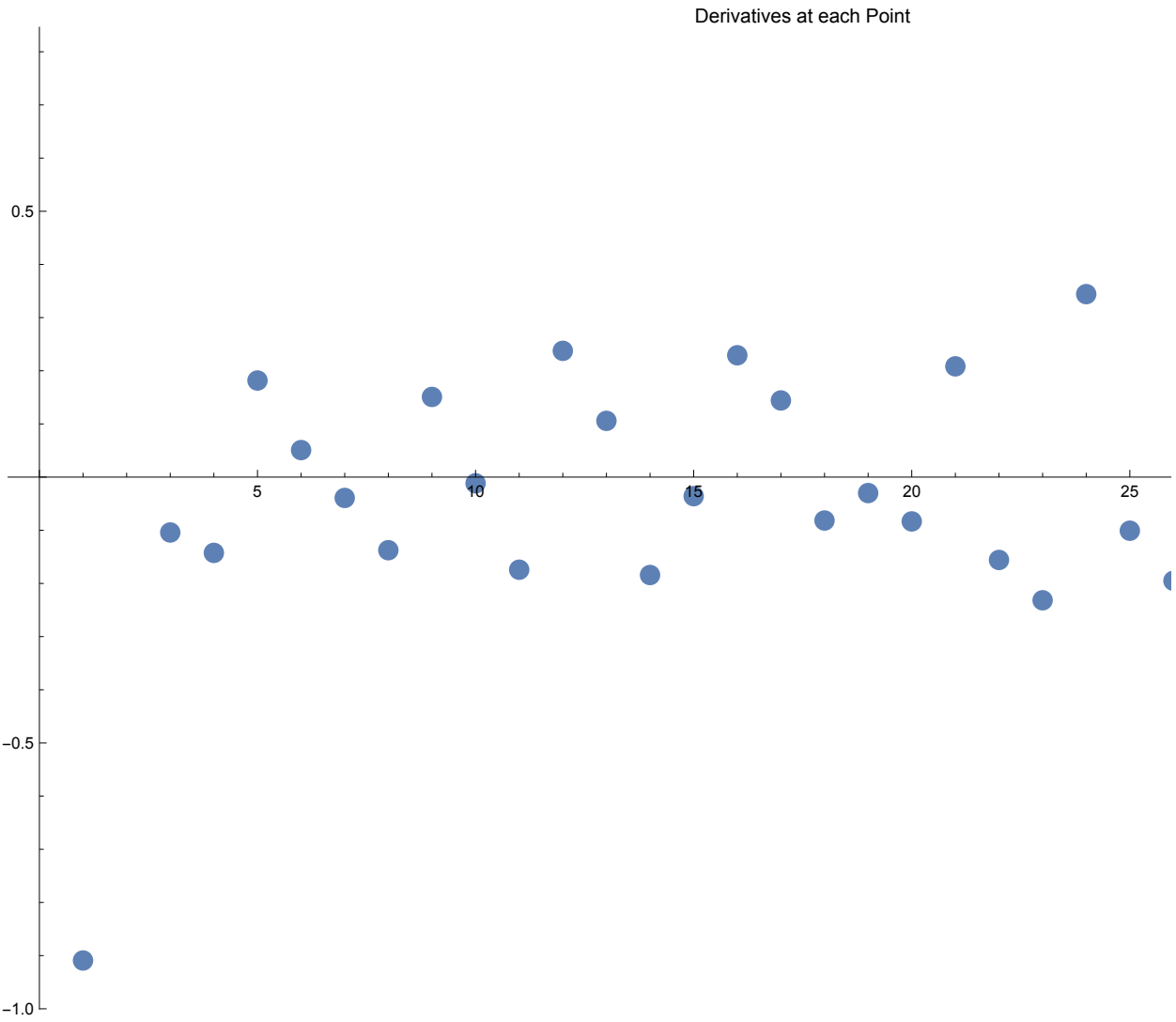
Out[680]= {0.31, 7.11535, 29.3497, 67.0131, 120.106, 188.627, 272.578, 371.957, 486.766, 617.003, 762.67, 923.765, 1100.29, 1292.24, 1499.63, 1722.44, 1960.68, 2214.35, 2483.45, 2767.98, 3067.93, 3383.32, 3714.13, 4060.38, 4422.05, 4799.15, 5191.68, 5599.64, 6023.03, 6461.85, 6916.1, 7385.77, 7870.88, 8371.41, 8887.38}

Error For Taylor Polynomial

In[864]:=
```
Tayerr  = Table[{}, n];
For[i = 1, i ≤ 35, i++,
 Tayerr[[i]] = Abs[fxi[[i]] - Taylorpol[xi[[i]]]]
]
Tayerr
m1 = Transpose[{xi, derivfxi}];
ListPlot[m1, PlotLabel → "Derivatives at each Point"]
ListPlot[Orderedpairs];
```

Out[866]=
$$\{1.38778 \times 10^{-15}, 6.79535, 28.7897, 66.8431, 119.746, 188.197, 272.118, 371.597, 486.496, 616.443, 762.42, 923.425, 1099.69, 1291.73, 1499.29, 1721.97, 1959.97, 2213.63, 2482.84, 2767.33, 3067.42, 3382.4, 3713.86, 4059.78, 4421.32, 4798.68, 5191.24, 5599.02, 6022.33, 6461.02, 6914.98, 7384.79, 7870.12, 8370.47, 8886.23\}$$

Out[868]=



Making the Lagrange Interpolating Polynomial

Making the Lagrange Interpolating Polynomial

In[687]:=
```
Spacing = xi[[2]] - xi[[1]];
LagVar = Table[{}, n];
```

In[689]:=
```
(*Making the divided difference table*)
MatrixForm[DivDif = Table[{}, n, n]];
For[i = 1, i ≤ n, i++,
 DivDif[[i, 1]] = fxi[[i]]]

For[ j = 2, j ≤ n, j++,
 For[i = j, i ≤ n, i++,
  DivDif[[i, j]] = (DivDif[[i, j - 1]] - DivDif[[i - 1, j - 1]]) /
     ((xi[[i]]) - (xi[[i - (j - 1)]]))]]
MatrixForm[DivDif];
```

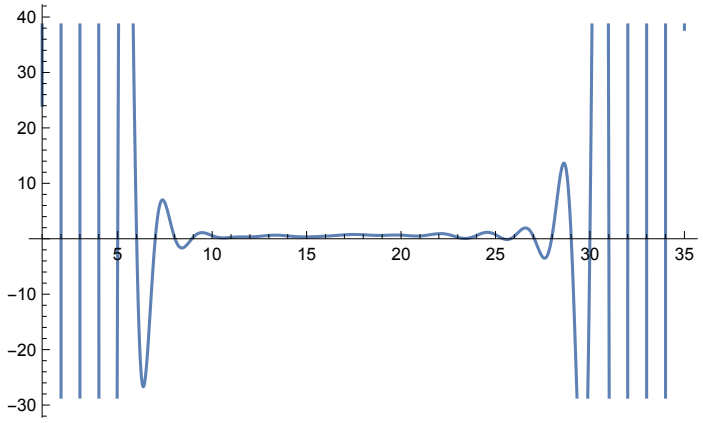
In[693]:=
```
LagPol[x_] := Sum[DivDif[[i, i]] * Product[(x - xi[[j]]), {j, 1, i - 1}], {i, 1, n}]
Simplify[LagPol[x]]
LagPol[xi]
```

Out[694]=
$$-1.14249\times10^{9}+4.68372\times10^{9}\,x-8.67843\times10^{9}\,x^{2}+9.79642\times10^{9}\,x^{3}-7.63548\times10^{9}\,x^{4}+4.40596\times10^{9}\,x^{5}-1.96804\times10^{9}\,x^{6}+7.01807\times10^{8}\,x^{7}-2.04348\times10^{8}\,x^{8}+4.94176\times10^{7}\,x^{9}-1.00573\times10^{7}\,x^{10}+1.74048\times10^{6}\,x^{11}-258\,242.\,x^{12}+33\,066.8\,x^{13}-3672.9\,x^{14}+355.317\,x^{15}-30.027\,x^{16}+2.22126\,x^{17}-0.144016\,x^{18}+0.00818699\,x^{19}-0.000407898\,x^{20}+0.0000177882\,x^{21}-6.77478\times10^{-7}\,x^{22}+2.24597\times10^{-8}\,x^{23}-6.45184\times10^{-10}\,x^{24}+1.59623\times10^{-11}\,x^{25}-3.37419\times10^{-13}\,x^{26}+6.03032\times10^{-15}\,x^{27}-8.98544\times10^{-17}\,x^{28}+1.0953\times10^{-18}\,x^{29}-1.06354\times10^{-20}\,x^{30}+7.90884\times10^{-23}\,x^{31}-4.2283\times10^{-25}\,x^{32}+1.4465\times10^{-27}\,x^{33}-2.37772\times10^{-30}\,x^{34}$$

Out[695]=
```
{0.31, 0.32, 0.56, 0.17, 0.36, 0.43, 0.46, 0.36, 0.27, 0.56, 0.25, 0.34, 0.6, 0.51,
 0.34, 0.47, 0.71, 0.72, 0.61, 0.65, 0.51, 0.92, 0.27, 0.6, 0.73, 0.47, 0.439999,
 0.619991, 0.699966, 0.829926, 1.11993, 0.978516, 0.759766, 0.933594, 1.13867}
```

In[696]:=
```
Plot[LagPol[x], {x, 1, n}]
```

Out[696]=



Lagrange Error

In[697]:=
```
LagErr = Table[{}, n];
For[i = 1, i ≤ 35, i++,
 LagErr[[i]] = Abs[fxi[[i]] - LagPol[xi[[i]]]]
]
LagErr
```

Out[699]=
$$\{0.,\ 0.,\ 0.,\ 1.38778\times10^{-16},\ 1.11022\times10^{-16},\ 2.77556\times10^{-16},\ 5.38458\times10^{-15},\ 3.07532\times10^{-14},\ 7.68274\times10^{-14},\ 1.71863\times10^{-13},\ 1.06581\times10^{-13},\ 2.59515\times10^{-13},\ 7.58282\times10^{-14},\ 8.90177\times10^{-13},\ 7.02355\times10^{-12},\ 2.02636\times10^{-11},\ 1.73168\times10^{-11},\ 7.43239\times10^{-11},\ 4.48926\times10^{-11},\ 6.13363\times10^{-10},\ 9.07457\times10^{-10},\ 8.61299\times10^{-9},\ 4.60248\times10^{-9},\ 5.57397\times10^{-8},\ 2.7083\times10^{-7},\ 1.16213\times10^{-7},\ 3.69298\times10^{-6},\ 0.0000134999,\ 0.0000346699,\ 0.0000745346,\ 0.0000682556,\ 0.00120289,\ 0.0000528598,\ 0.00746221,\ 0.0124535\}$$

Derivatives according to Lagrange Interpolation

In[700]:=
```
LagDer[x_] = Simplify[D[LagPol[x], x]]
LagDer[xi]
```

Out[700]=
$$4.68372\times10^{9} - 1.73569\times10^{10}\,x + 2.93893\times10^{10}\,x^2 - 3.05419\times10^{10}\,x^3 + 2.20298\times10^{10}\,x^4 - 1.18082\times10^{10}\,x^5 + 4.91265\times10^{9}\,x^6 - 1.63478\times10^{9}\,x^7 + 4.44759\times10^{8}\,x^8 - 1.00573\times10^{8}\,x^9 + 1.91453\times10^{7}\,x^{10} - 3.0989\times10^{6}\,x^{11} + 429\,868.\,x^{12} - 51\,420.6\,x^{13} + 5329.75\,x^{14} - 480.432\,x^{15} + 37.7614\,x^{16} - 2.59229\,x^{17} + 0.155553\,x^{18} - 0.00815796\,x^{19} + 0.000373553\,x^{20} - 0.0000149045\,x^{21} + 5.16573\times10^{-7}\,x^{22} - 1.54844\times10^{-8}\,x^{23} + 3.99057\times10^{-10}\,x^{24} - 8.77289\times10^{-12}\,x^{25} + 1.62819\times10^{-13}\,x^{26} - 2.51592\times10^{-15}\,x^{27} + 3.17636\times10^{-17}\,x^{28} - 3.19063\times10^{-19}\,x^{29} + 2.45174\times10^{-21}\,x^{30} - 1.35306\times10^{-23}\,x^{31} + 4.77346\times10^{-26}\,x^{32} - 8.08425\times10^{-29}\,x^{33}$$

Out[701]=
$$\{3.42631\times10^{7},\ -1.06064\times10^{6},\ 67\,869.3,\ -6743.31,\ 926.679,\ -165.477,\ 36.625,\ -10.,\ 12.,\ 0.,\ 0.,\ 0.,\ 2048.,\ -8192.,\ -32\,768.,\ 65\,536.,\ 131\,072.,\ 1.57286\times10^{6},\ -4.1943\times10^{6},\ 1.04858\times10^{7},\ 1.67772\times10^{7},\ -2.51658\times10^{7},\ -1.67772\times10^{7},\ 3.35544\times10^{7},\ 6.0398\times10^{8},\ 5.36871\times10^{8},\ 3.22123\times10^{9},\ -3.7581\times10^{9},\ -8.58993\times10^{9},\ 2.14748\times10^{9},\ 1.50324\times10^{10},\ -1.28849\times10^{10},\ 6.01295\times10^{10},\ 3.43597\times10^{10},\ 5.15396\times10^{10}\}$$

Creating the Spline

In[702]:=
```
d = Length[xi];
hi = Table[{}, Length[xi] - 1];
For[i = 1, i ≤ Length[xi] - 1, i++,
 hi[[i]] = xi[[i + 1]] - xi[[i]]]
hi;
(*Creating a Matrix to put our known variables in*)
aMatrix = Table[{}, d - 2, d - 2];
MatrixForm[aMatrix];

For[ j = 1, j ≤ d - 2, j++,
 For[i = 1, i ≤ d - 2, i++,
  aMatrix[[i, j]] = 0
 ]
]

(*First diagonal (i)*)
For[ j = 2, j ≤ d - 2, j++,
 For[i = j - 1, i ≤ j - 1, i++,
  aMatrix[[i, j]] = hi[[i + 1]]
 ]
]
```

```
(*Middle diagonal (i)*)
For[ j = 1, j ≤ d - 2, j++,
 For[i = j, i ≤ j, i++,
  aMatrix[[i, j]] = (2 (hi[[i]] + hi[[i + 1]]))
 ]
]
(*Bottom Diagonal (i-1)*)
For[ j = 1, j ≤ d - 3, j++,
 For[i = j + 1, i ≤ j + 1, i++,
  aMatrix[[i, j]] = hi[[i]]
 ]
]


(*Matrix A to solve for cj*)
MatrixForm[aMatrix];
(*Matrix b to solve for cj's*)
bMatrix = Table[{}, d - 2, 1];
For[i = 2, i ≤ d - 1, i++,
 bMatrix[[i - 1]] =
  ((3 / hi[[i]]) (fxi[[i + 1]] - fxi[[i]])) - ((3 / hi[[i - 1]]) (fxi[[i]] - fxi[[i - 1]]))
]

(*Correct Matrix for b*)
MatrixForm[bMatrix];
cj = LinearSolve[aMatrix, bMatrix];
(*Do not redo this Function or else to many variables in cj*)
PrependTo[cj, 0];
AppendTo[cj, 0];
cj;
(*Creating the bj Mat*)
bj = Table[{}, d - 1, 1];
MatrixForm[bj];
(*Filling bj matrix with proper values*)
For[i = 1, i ≤ d - 1, i++,
 bj[[i]] =
  ((fxi[[i + 1]] - fxi[[i]]) / hi[[i]]) - ((hi[[i]] / 3) (2 cj[[i]] + cj[[i + 1]]))
]
bj;
(*Making a Matrix for dj*)
dj = Table[{}, d - 1, 1];
MatrixForm[dj];
For[i = 1, i ≤ d - 1, i++,
```

```
 dj[[i]] = (1 / (3 hi[[i]])) (cj[[i + 1]] - cj[[i]])
]
dj;
```

In[728]:=
```
SplineFuncs3 = Table[{}, d - 1];
For[i = 1, i ≤ d - 1, i++,
 SplineFuncs3[[i]] =
  fxi[[i]] + bj[[i]] (x - xi[[i]]) + cj[[i]] (x - xi[[i]])^2 + dj[[i]] (x - xi[[i]])^3
]
MatrixForm[SplineFuncs3];
Remove[Wild2]

Wild2[x_] := Piecewise[ {
   {SplineFuncs3[[1]], xi[[1]] ≤ x ≤ xi[[2]]},
   {SplineFuncs3[[2]], xi[[2]] ≤ x ≤ xi[[3]]},
   {SplineFuncs3[[3]], xi[[3]] ≤ x ≤ xi[[4]]},
   {SplineFuncs3[[4]], xi[[4]] ≤ x ≤ xi[[5]]},
   {SplineFuncs3[[5]], xi[[5]] ≤ x ≤ xi[[6]]},
   {SplineFuncs3[[6]], xi[[6]] ≤ x ≤ xi[[7]]},
   {SplineFuncs3[[7]], xi[[7]] ≤ x ≤ xi[[8]]},
   {SplineFuncs3[[8]], xi[[8]] ≤ x ≤ xi[[9]]},
   {SplineFuncs3[[9]], xi[[9]] ≤ x ≤ xi[[10]]},
   {SplineFuncs3[[10]], xi[[10]] ≤ x ≤ xi[[11]]},
   {SplineFuncs3[[11]], xi[[11]] ≤ x ≤ xi[[12]]},
   {SplineFuncs3[[12]], xi[[12]] ≤ x ≤ xi[[13]]},
   {SplineFuncs3[[13]], xi[[13]] ≤ x ≤ xi[[14]]},
   {SplineFuncs3[[14]], xi[[14]] ≤ x ≤ xi[[15]]},
   {SplineFuncs3[[15]], xi[[15]] ≤ x ≤ xi[[16]]},
   {SplineFuncs3[[16]], xi[[16]] ≤ x ≤ xi[[17]]},
   {SplineFuncs3[[17]], xi[[17]] ≤ x ≤ xi[[18]]},
   {SplineFuncs3[[18]], xi[[18]] ≤ x ≤ xi[[19]]},
   {SplineFuncs3[[19]], xi[[19]] ≤ x ≤ xi[[20]]},
   {SplineFuncs3[[20]], xi[[20]] ≤ x ≤ xi[[21]]},
   {SplineFuncs3[[21]], xi[[21]] ≤ x ≤ xi[[22]]},
   {SplineFuncs3[[22]], xi[[22]] ≤ x ≤ xi[[23]]},
   {SplineFuncs3[[23]], xi[[23]] ≤ x ≤ xi[[24]]},
   {SplineFuncs3[[24]], xi[[24]] ≤ x ≤ xi[[25]]},
   {SplineFuncs3[[25]], xi[[25]] ≤ x ≤ xi[[26]]},
   {SplineFuncs3[[26]], xi[[26]] ≤ x ≤ xi[[27]]},
   {SplineFuncs3[[27]], xi[[27]] ≤ x ≤ xi[[28]]},
   {SplineFuncs3[[28]], xi[[28]] ≤ x ≤ xi[[29]]},
   {SplineFuncs3[[29]], xi[[29]] ≤ x ≤ xi[[30]]},
   {SplineFuncs3[[30]], xi[[30]] ≤ x ≤ xi[[31]]},
   {SplineFuncs3[[31]], xi[[31]] ≤ x ≤ xi[[32]]},
```

```
    {SplineFuncs3[[32]], xi[[32]] ≤ x ≤ xi[[33]]},
    {SplineFuncs3[[33]], xi[[33]] ≤ x ≤ xi[[34]]},
    {SplineFuncs3[[34]], xi[[34]] ≤ x ≤ xi[[35]]}
   }]
```

Creating the Hermite

In[733]:=
```
m = 2 * Length[xi];
MatrixForm[DivDif2 = Table[{}, m, m + 1]];

For[i = 1, i ≤ m/2, i++,
 DivDif2[[(2 i), 1]] = xi[[i]]]
For[i = 1, i ≤ m/2, i++,
 DivDif2[[(2 i) - 1, 1]] = xi[[i]]]

For[i = 1, i ≤ m / 2, i++,
 DivDif2[[(2 i), 2]] = fxi[[i]]]
For[i = 1, i ≤ m/2, i++,
 DivDif2[[(2 i) - 1, 2]] = fxi[[i]]]

For[ j = 2, j ≤ 2, j++,
 For[i = j, i ≤ m, i++,
  DivDif2[[i, j + 1]] = (DivDif2[[i, j]] - DivDif2[[i - 1, j]]) /
    (DivDif2[[i, j - 1]] - DivDif2[[i - 1, j - 1]])
]]
MatrixForm[DivDif2];
```

Power: Infinite expression $\frac{1}{0}$ encountered.

Infinity: Indeterminate expression 0. ComplexInfinity encountered.

Power: Infinite expression $\frac{1}{0}$ encountered.

Infinity: Indeterminate expression 0. ComplexInfinity encountered.

Power: Infinite expression $\frac{1}{0}$ encountered.

General: Further output of Power::infy will be suppressed during this calculation.

Infinity: Indeterminate expression 0. ComplexInfinity encountered.

General: Further output of Infinity::indet will be suppressed during this calculation.

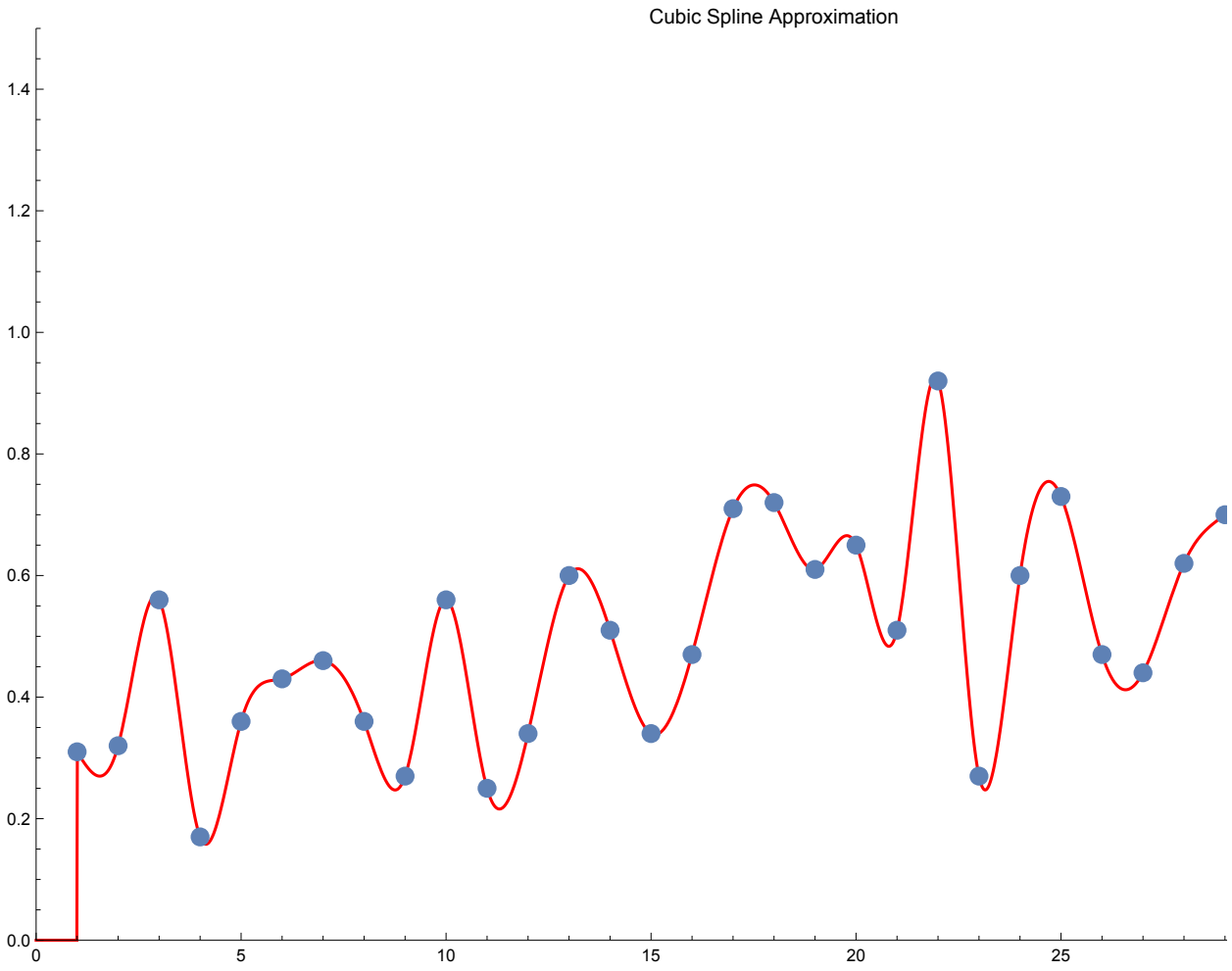
In[741]:=
```
h = xi[[2]] - xi[[1]];
derivfxi = Table[{}, n];

For[i = 3, i ≤ n - 2, i++,
 derivfxi[[i]] =
  (fxi[[i - 2]] - 8 fxi[[i - 1]] +
     8 fxi[[i + 1]] - fxi[[i + 2]]) / (12 h)]
For[i = 1, i ≤ 2, i++,
 derivfxi[[i]] =
  (-25 fxi[[i]] + 48 fxi[[i + 1]] - 36 fxi[[i + 2]] +
     16 fxi[[i + 3]] - 3 fxi[[i + 4]]) (1 / (12 h))]
For[i = n, i ≥ n - 1, i--,
 derivfxi[[i]] =
  (-25 fxi[[i]] + 48 fxi[[i - 1]] - 36 fxi[[i - 2]] +
     16 fxi[[i - 3]] - 3 fxi[[i - 4]]) (1 / (12 (-h)))]
derivfxi;

For[i = 1, i ≤ m / 2, i++,
 DivDif2[[(2 i), 3]] = derivfxi[[i]]]
MatrixForm[DivDif2];

For[ j = 3, j ≤ m, j++,
 For[i = j, i ≤ m, i++, DivDif2[[i, j + 1]] = (DivDif2[[i, j]] - DivDif2[[i - 1, j]]) /
    (DivDif2[[i, j - (j - 1)]] - DivDif2[[i - (j - 1), j - (j - 1)]])]]
MatrixForm[DivDif2];

Herm[x_] :=
 Sum[DivDif2[[i, i + 1]] * Product[(x - DivDif2[[j, 1]]), {j, 1, i - 1}], {i, 1, m}]
Simplify[Herm[x]]
```

Out[752]=
$$1.0117\times10^{17}-8.25282\times10^{17}\,x+3.20273\times10^{18}\,x^2-7.90144\times10^{18}\,x^3+1.39695\times10^{19}\,x^4-1.89125\times10^{19}\,x^5+2.04553\times10^{19}\,x^6-1.82043\times10^{19}\,x^7+1.36242\times10^{19}\,x^8-8.71943\times10^{18}\,x^9+4.83588\times10^{18}\,x^{10}-2.34938\times10^{18}\,x^{11}+1.00878\times10^{18}\,x^{12}-3.85705\times10^{17}\,x^{13}+1.32159\times10^{17}\,x^{14}-4.08032\times10^{16}\,x^{15}+1.14051\times10^{16}\,x^{16}-2.89806\times10^{15}\,x^{17}+6.71864\times10^{14}\,x^{18}-1.42561\times10^{14}\,x^{19}+2.7764\times10^{13}\,x^{20}-4.97504\times10^{12}\,x^{21}+8.22042\times10^{11}\,x^{22}-1.2549\times10^{11}\,x^{23}+1.77289\times10^{10}\,x^{24}-2.32141\times10^{9}\,x^{25}+2.82087\times10^{8}\,x^{26}-3.18458\times10^{7}\,x^{27}+3.34323\times10^{6}\,x^{28}-326\,633.\,x^{29}+29\,716.1\,x^{30}-2518.57\,x^{31}+198.913\,x^{32}-14.6405\,x^{33}+1.00413\,x^{34}-0.0641547\,x^{35}+0.00381614\,x^{36}-0.000211156\,x^{37}+0.0000108551\,x^{38}-5.17564\times10^{-7}\,x^{39}+2.28336\times10^{-8}\,x^{40}-9.29013\times10^{-10}\,x^{41}+3.46935\times10^{-11}\,x^{42}-1.18076\times10^{-12}\,x^{43}+3.62097\times10^{-14}\,x^{44}-9.80634\times10^{-16}\,x^{45}+2.25009\times10^{-17}\,x^{46}-3.9051\times10^{-19}\,x^{47}+2.61056\times10^{-21}\,x^{48}+1.56475\times10^{-22}\,x^{49}-9.38003\times10^{-24}\,x^{50}+3.38943\times10^{-25}\,x^{51}-9.74978\times10^{-27}\,x^{52}+2.39342\times10^{-28}\,x^{53}-5.14966\times10^{-30}\,x^{54}+9.82507\times10^{-32}\,x^{55}-1.67012\times10^{-33}\,x^{56}+2.53149\times10^{-35}\,x^{57}-3.41535\times10^{-37}\,x^{58}+4.08512\times10^{-39}\,x^{59}-4.30601\times10^{-41}\,x^{60}+3.96705\times10^{-43}\,x^{61}-3.15955\times10^{-45}\,x^{62}+2.14407\times10^{-47}\,x^{63}-1.21564\times10^{-49}\,x^{64}+5.60368\times10^{-52}\,x^{65}-2.01773\times10^{-54}\,x^{66}+5.32409\times10^{-57}\,x^{67}-9.15678\times10^{-60}\,x^{68}+7.70308\times10^{-63}\,x^{69}$$

In[753]:=

In[857]:=
```
Show[
 Plot[Wild2[x], {x, 0, 36}, PlotLabel → "Cubic Spline Approximation",
  PlotStyle → Red, PlotRange → {{0, 36}, {0, 1.5}}, PlotLegends → "Cubic Spline"],
 ListPlot[Orderedpairs]
]
```

Out[857]=



Project 8 code for Project 6

In[755]:=
```
howmany3 = n;
hstep3 = (xi[[n]] - xi[[1]]) / (howmany3 - 1);
points3 = howmany3;
xip8c = Table[ xi[[i]], {i, 1, points3 }];
fxip8c = Table[{}, {points3 }];
For[i = 1, i ≤ points3, i++,
 fxip8c[[i]] = fxi[[i]]
]
fxip8c;

TrapInt3 = ( fxi[[1]]
      + fxi[[n]]
      + 2 * Sum[fxi[[i + 1]], {i, 1, points3 - 2}]) (hstep3 / 2);
N[TrapInt3]
```

Out[763]=
```
19.31
```

In[764]:=
```
howmany4 = n;
hstep4 = (xi[[n]] - xi[[1]]) / (howmany4 - 1);
points4 = howmany4;
xip8d = Table[ xi[[i]], {i, 1, points4 }];
```

In[768]:=
```
fxip8d = Table[{}, {points4 }];
For[i = 1, i ≤ points4, i++,
 fxip8d[[i]] = fxi[[i]]
]
fxip8d;
TrapSimp4 = ( fxip8d[[1]]
      + fxip8d[[points4]]
      + 2 * Sum[fxip8d[[2 i - 1]], {i, 2, (points4 - 1) / 2}]
      + 4 * Sum[fxip8d[[2 i]], {i, 1, (points4 - 1) / 2}]) (hstep4 / 3);
N[TrapSimp4]
```

Out[772]=
```
19.4667
```

Using what we Learned from Gaussian Quadrature to pick points for the Lagrange and Hermite

In[773]:=
```
xi2 = {2, 4, 8, 17, 25, 30, 33, 35}
fxi2 = {0.32, 0.17, 0.36, 0.71, 0.73, 0.83, 0.76, 1.15}
```

Out[773]= {2, 4, 8, 17, 25, 30, 33, 35}

Out[774]= {0.32, 0.17, 0.36, 0.71, 0.73, 0.83, 0.76, 1.15}

Making the Lagrange Interpolating Polynomial with better points

In[775]:=
```
n2 = Length[xi2];
LagVar2 = Table[{}, n2];
```

In[777]:=
```
(*Making the divided difference table*)
MatrixForm[DivDif21 = Table[{}, n2, n2]]
For[i = 1, i ≤ n2, i++,
 DivDif21[[i, 1]] = fxi2[[i]]]

For[ j = 2, j ≤ n2, j++,
 For[i = j, i ≤ n2, i++,
  DivDif21[[i, j]] = (DivDif21[[i, j - 1]] - DivDif21[[i - 1, j - 1]]) /
    ((xi2[[i]]) - (xi2[[i - (j - 1)]]))]]
MatrixForm[DivDif21]
```

Out[777]//MatrixForm=

$$\begin{pmatrix} \{\} & \{\} & \{\} & \{\} & \{\} & \{\} & \{\} & \{\} \\ \{\} & \{\} & \{\} & \{\} & \{\} & \{\} & \{\} & \{\} \\ \{\} & \{\} & \{\} & \{\} & \{\} & \{\} & \{\} & \{\} \\ \{\} & \{\} & \{\} & \{\} & \{\} & \{\} & \{\} & \{\} \\ \{\} & \{\} & \{\} & \{\} & \{\} & \{\} & \{\} & \{\} \\ \{\} & \{\} & \{\} & \{\} & \{\} & \{\} & \{\} & \{\} \\ \{\} & \{\} & \{\} & \{\} & \{\} & \{\} & \{\} & \{\} \\ \{\} & \{\} & \{\} & \{\} & \{\} & \{\} & \{\} & \{\} \end{pmatrix}$$

Out[780]//MatrixForm=

$$\begin{pmatrix} 0.32 & \{\} & \{\} & \{\} & \{\} & \{\} & \{\} \\ 0.17 & -0.075 & \{\} & \{\} & \{\} & \{\} & \{\} \\ 0.36 & 0.0475 & 0.0204167 & \{\} & \{\} & \{\} & \{\} \\ 0.71 & 0.0388889 & -0.000662393 & -0.00140527 & \{\} & \{\} & \{\} \\ 0.73 & 0.0025 & -0.00214052 & -0.0000703871 & 0.0000580384 & \{\} & \{\} \\ 0.83 & 0.02 & 0.00134615 & 0.000158485 & 8.80279\times10^{-6} & -1.75842\times10^{-6} & \{\} \\ 0.76 & -0.0233333 & -0.00541667 & -0.000422676 & -0.0000232465 & -1.10515\times10^{-6} & 2.10732\times \\ 1.15 & 0.195 & 0.0436667 & 0.00490833 & 0.000296167 & 0.0000118301 & 4.17267\times \end{pmatrix}$$

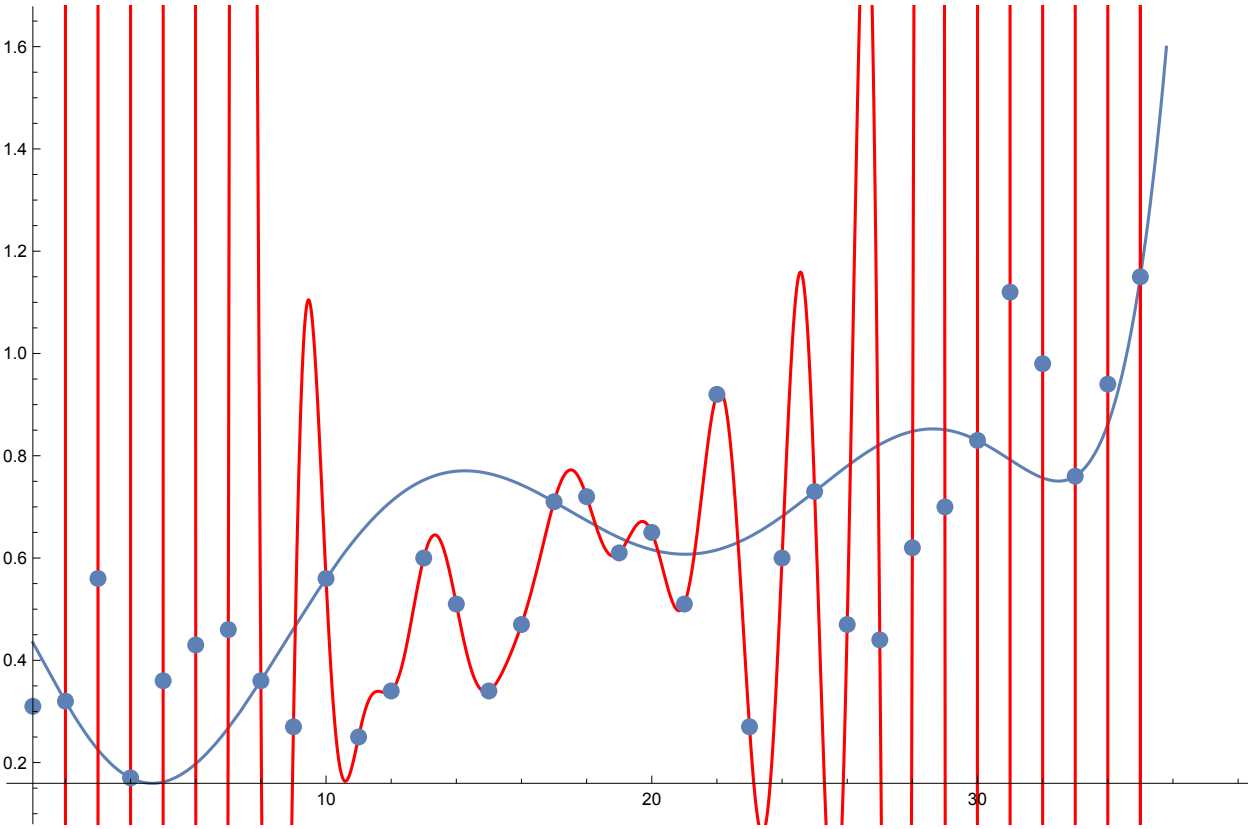
In[781]:=
```
LagPol2[x_] := Sum[DivDif21[[i, i]] * Product[(x - xi2[[j]]), {j, 1, i - 1}], {i, 1, n2}]
Simplify[LagPol2[x]]
LagPol[xi2]
```

Out[782]=
$$0.528147 - 0.063653\,x - 0.0420863\,x^2 + 0.0134154\,x^3 - 0.00136017\,x^4 + 0.0000635061\,x^5 - 1.40763 \times 10^{-6}\,x^6 + 1.20059 \times 10^{-8}\,x^7$$

Out[783]=
```
{0.32, 0.17, 0.36, 0.71, 0.73, 0.829926, 0.759766, 1.13867}
```

In[784]:=
```
Show[
 Plot[LagPol2[x], {x, 1, n + 5}],
 Plot[LagPol[x], {x, 1, n + 5}, PlotStyle → Red],
 ListPlot[Orderedpairs]
]
```

Out[784]=



In[785]:=
```
LagPol[36]
LagPol2[36]
```

Out[785]=
$$-1.71802 \times 10^9$$

Out[786]=
```
1.75192
```

Creating the Hermite

In[822]:=
```
xi3 = {4, 8, 12, 16, 20, 24, 28, 32, 36}
fxi3 = {0.17, 0.36 , 0.34, 0.47, 0.65, 0.6, 0.62, 0.98, 1.45}
```

Out[822]= {4, 8, 12, 16, 20, 24, 28, 32, 36}

Out[823]= {0.17, 0.36, 0.34, 0.47, 0.65, 0.6, 0.62, 0.98, 1.45}

In[824]:=
```
w = Length[xi3]
m2 = 2 * Length[xi3];
MatrixForm[DivDif3 = Table[{}, m2, m2 + 1]];

For[i = 1, i ≤ m2 / 2, i++,
 DivDif3[[(2 i), 1]] = xi3[[i]]]
For[i = 1, i ≤ m2 / 2, i++,
 DivDif3[[(2 i) - 1, 1]] = xi3[[i]]]

For[i = 1, i ≤ m2 / 2, i++,
 DivDif3[[(2 i), 2]] = fxi3[[i]]]
For[i = 1, i ≤ m2 / 2, i++,
 DivDif3[[(2 i) - 1, 2]] = fxi3[[i]]]

For[ j = 2, j ≤ 2, j++,
 For[i = j, i ≤ m2, i++,
  DivDif3[[i, j + 1]] = (DivDif3[[i, j]] - DivDif3[[i - 1, j]]) /
    (DivDif3[[i, j - 1]] - DivDif3[[i - 1, j - 1]])
]]
MatrixForm[DivDif3];
```

Out[824]= 9

Power: Infinite expression $\frac{1}{0}$ encountered.

Infinity: Indeterminate expression 0. ComplexInfinity encountered.

Power: Infinite expression $\frac{1}{0}$ encountered.

Infinity: Indeterminate expression 0. ComplexInfinity encountered.

Power: Infinite expression $\frac{1}{0}$ encountered.

General: Further output of Power::infy will be suppressed during this calculation.

Infinity: Indeterminate expression 0. ComplexInfinity encountered.

General: Further output of Infinity::indet will be suppressed during this calculation.

In[833]:=
```
h2 = xi3[[2]] - xi3[[1]];
derivfxi3 = Table[{}, w]

For[i = 3, i ≤ w, i++,
 derivfxi3[[i]] =
  (fxi3[[i - 2]] - 8 fxi3[[i - 1]] +
     8 fxi3[[i + 1]] - fxi3[[i + 2]]) / (12 h2)]
For[i = 1, i ≤ 2, i++,
 derivfxi3[[i]] =
  (-25 fxi3[[i]] + 48 fxi3[[i + 1]] - 36 fxi3[[i + 2]] +
     16 fxi3[[i + 3]] - 3 fxi3[[i + 4]]) (1 / (12 h2))]
For[i = w, i ≥ w - 1, i--,
 derivfxi3[[i]] =
  (-25 fxi3[[i]] + 48 fxi3[[i - 1]] - 36 fxi3[[i - 2]] +
     16 fxi3[[i - 3]] - 3 fxi3[[i - 4]]) (1 / (12 (-h2)))]
derivfxi3

For[i = 1, i ≤ m2 / 2, i++,
 DivDif3[[(2 i), 3]] = derivfxi3[[i]]]
MatrixForm[DivDif3];

For[ j = 3, j ≤ m2, j++,
 For[i = j, i ≤ m2, i++, DivDif3[[i, j + 1]] = (DivDif3[[i, j]] - DivDif3[[i - 1, j]]) /
    (DivDif3[[i, j - (j - 1)]] - DivDif3[[i - (j - 1), j - (j - 1)]])]]
MatrixForm[DivDif3];

Herm2[x_] :=
 Sum[DivDif3[[i, i + 1]] * Product[(x - DivDif3[[j, 1]]), {j, 1, i - 1}], {i, 1, m2}]
Simplify[Herm2[x]]
Herm2[xi3]
```

Out[834]=
```
{{}, {}, {}, {}, {}, {}, {}, {}, {}}
```

Part: Part 10 of {0.17, 0.36, 0.34, 0.47, 0.65, 0.6, 0.62, 0.98, 1.45} does not exist.

Part: Part 10 of {0.17, 0.36, 0.34, 0.47, 0.65, 0.6, 0.62, 0.98, 1.45} does not exist.

Part: Part 11 of {0.17, 0.36, 0.34, 0.47, 0.65, 0.6, 0.62, 0.98, 1.45} does not exist.

General: Further output of Part::partw will be suppressed during this calculation.

Out[838]=
```
{0.1325, -0.0208333, 0.00833333, 0.0466667,
 0.0158333, -0.015625, 0.0466667, 0.153125, 0.0808333}
```

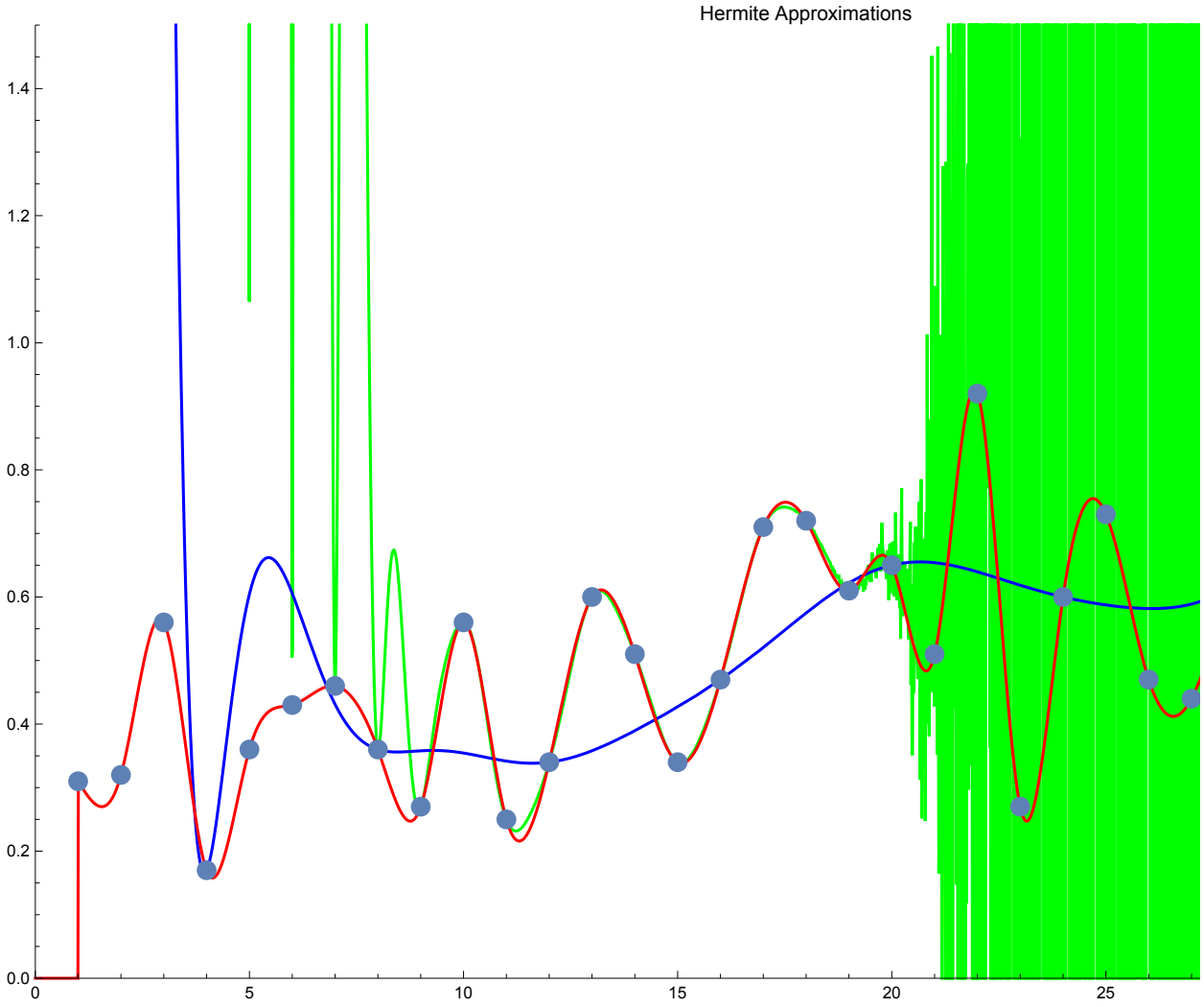
Out[844]= $975.205 - 1282.95\,x + 753.217\,x^2 - 262.819\,x^3 + 61.2419\,x^4 - 10.1433\,x^5 + 1.23969\,x^6 - 0.114445\,x^7 + 0.00809368\,x^8 - 0.000441605\,x^9 + 0.0000186072\,x^{10} - 6.02321 \times 10^{-7}\,x^{11} + 1.47944 \times 10^{-8}\,x^{12} - 2.69734 \times 10^{-10}\,x^{13} + 3.51874 \times 10^{-12}\,x^{14} - 3.08424 \times 10^{-14}\,x^{15} + 1.61191 \times 10^{-16}\,x^{16} - 3.74046 \times 10^{-19}\,x^{17}$

Out[845]= {0.17, 0.36, 0.34, 0.47, 0.65, 0.6, 0.62, 0.98, 1.45}

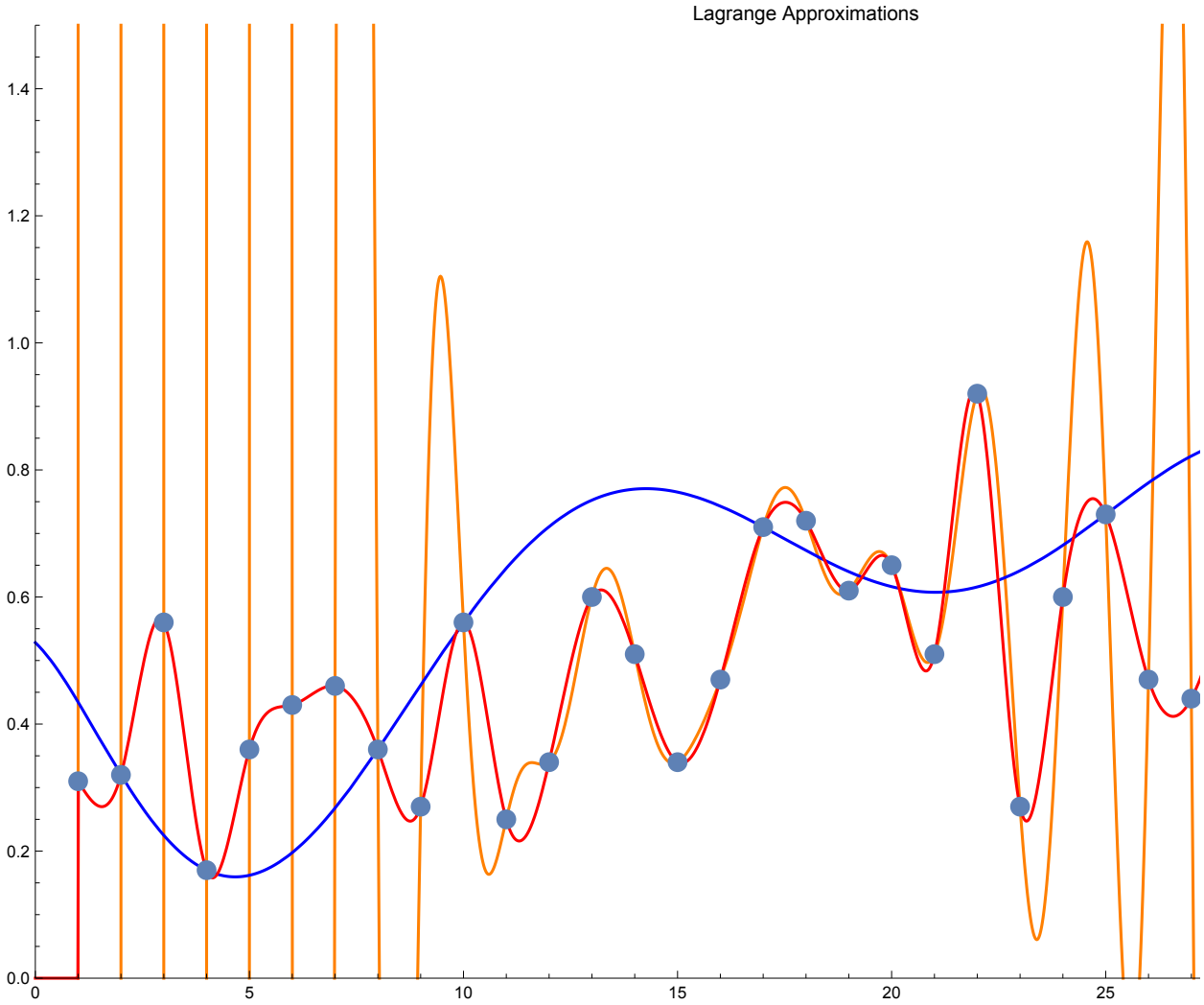
In[811]:=

In[854]:=
```
Show[
 Plot[{Herm[x]}, {x, 0, 36},
  PlotLegends → {"Hermite all points"}, PlotLabel → "Hermite Approximations",
  PlotRange → {{0, 36}, {0, 1.5}}, PlotStyle → Green],
 Plot[{Herm2[x]}, {x, 0, 36}, PlotLegends → {"Hermite select points"},
  PlotRange → {{0, 36}, {0, 1.5}}, PlotStyle → Blue],
 Plot[Wild2[x], {x, 0, 36}, PlotLegends → Automatic,
  PlotStyle → Red, PlotRange → {{0, 36}, {0, 1.5}}],
 ListPlot[Orderedpairs]
]
Show[
 Plot[{LagPol[x]}, {x, 0, 36},
  PlotLegends → {"Lagrange all points"}, PlotLabel → "Lagrange Approximations",
  PlotRange → {{0, 36}, {0, 1.5}}, PlotStyle → Orange],
 Plot[{LagPol2[x]}, {x, 0, 36}, PlotLegends → {"Lagrange select points"},
  PlotRange → {{0, 36}, {0, 1.5}}, PlotStyle → Blue],
 Plot[Wild2[x], {x, 0, 36}, PlotLegends → Automatic,
  PlotStyle → Red, PlotRange → {{0, 36}, {0, 1.5}}],
 ListPlot[Orderedpairs]
]
```

Out[854]=



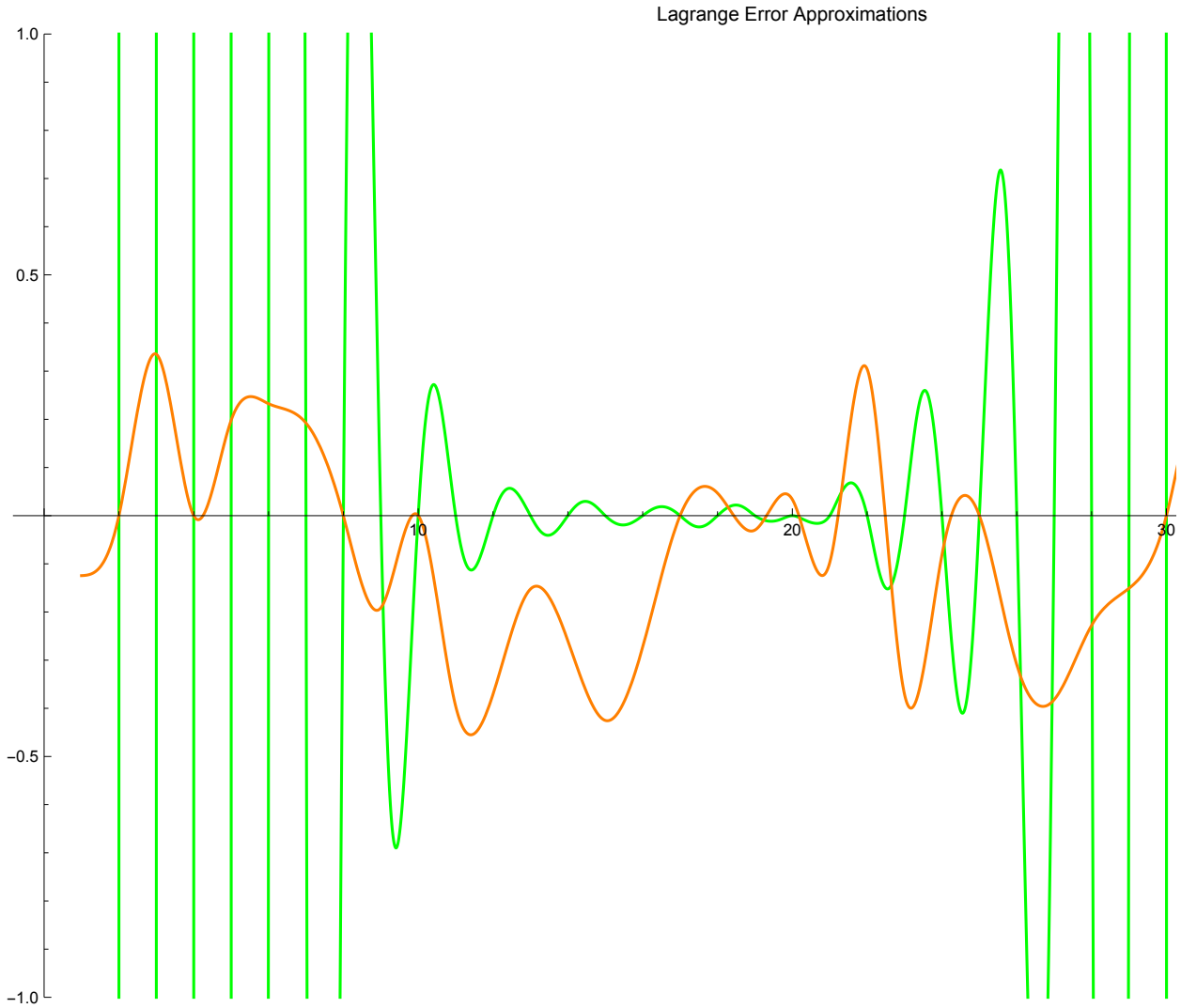
Out[855]=



Error for Lagrange

In[886]:=
```
Show[
 Plot[Wild2[x] - LagPol[x], {x, 1, n + 5},
  PlotRange → {-1, 1}, PlotLegends → {"Lagrange All points"},
  PlotLabel → "Lagrange Error Approximations", PlotStyle → Green],
 Plot[Wild2[x] - LagPol2[x], {x, 1, n + 5}, PlotLegends → {"Lagrange Select points"},
  PlotLabel → "Lagrange Error Approximations", PlotStyle → Orange]
]
```

Out[886]=



Error For Hermite

In[880]:=
```
Show[
 Plot[Wild2[x] - Herm[x], {x, 1, n + 5},
  PlotRange → {-1, 1}, PlotLegends → {"Hermite all points"},
  PlotLabel → "Hermite Error Approximations", PlotStyle → Green],
 Plot[Wild2[x] - Herm2[x], {x, 1, n + 5}, PlotLegends → {"Hermite select points"},
  PlotLabel → "Hermite Error Approximations", PlotStyle → Red]
]
```

Out[880]=