5 Linear and Nonlinear Homotopy

5-1 Introduction

Most often, there exist a fundamental task of solving systems of equations in geodesy. In such cases, many geodetic problems are represented as systems of multivariate polynomials. Frequently the bottleneck in solving such systems is to find out proper initial starting values for iterative methods, which lead to convergence to solutions with physical meaning. Although symbolic methods such as Groebner basis or resultants have been shown to be very efficient, providing solutions for systems of only modest size. In addition, in some cases the nonlinear equations representing the geodetic problems are not polynomial ones. Therefore, in case of some real life problems one should employ global numerical methods to avoid initial value problem. Using numerical algorithms to solve polynomial systems with tools originating from algebraic geometry is the main activity in the so called Numerical Algebraic Geometry. This is a new developing field on the crossroads of algebraic geometry, numerical analysis, computer science and engineering. Homotopy continuation method is a global numerical method to solve not only polynomial systems, but also nonlinear system in general. A very comprehensive summary of the literature of this subject can be found in Dickenstein and Emiris (2005) as well as in Palacz et al (2009)

5-2 Definition and basic concepts

The continuous deformation of an object to an other object is known as homotopy. Let us consider a simple geometric example. Define the homotopy between a circle and a square. The circle parametric equations,

\begin{align*}
x &= R \cos(\alpha) \\
y &= R \sin(\alpha)
\end{align*}

```
In[1]:= Clear["Global`*"]
In[2]:= circle = Table[{Cos[α], Sin[α]}, {α, 0, 2 Pi, 0.02}];
In[11]:= ListPlot[circle, Joined -> True, AspectRatio -> 1, ImageSize -> 200]
```

while the parametric equations of the square

\begin{align*}
x &= f(\alpha) R \cos(\alpha) \\
y &= f(\alpha) R \sin(\alpha)
\end{align*}

```
```
where

\[ f(\alpha) = \frac{1}{\max(|\sin(\alpha)|, |\cos(\alpha)|)} \]

In[4]:= square = Table[{Cos[\alpha], Sin[\alpha]} (1 / Max[Abs[Sin[\alpha]], Abs[Cos[\alpha]]]), {\alpha, 0, 2 Pi, 0.02}];
In[10]:= ListPlot[square, Joined -> True, AspectRatio -> 1, ImageSize -> 200]
Out[10]=

Now the homotopy function

\[ H(\alpha, \lambda) = \lambda \left( \frac{R \cos(\alpha)}{R \sin(\alpha)} \right) + (1 - \lambda) \left( \frac{f(\alpha) R \cos(\alpha)}{f(\alpha) R \sin(\alpha)} \right) \]

In geometric terms, the homotopy \( H \) provides us a continuous, smooth deformation from \( \text{square} \) - which is obtained for \( \lambda = 0 \) by \( H(\alpha, 0) \) - to \( \text{circle} \) - which is obtained for \( \lambda = 1 \) by \( H(\alpha, 1) \). The animation of the homotopy can be seen in the Fig 5.1 for \( \lambda \in [0, 1] \). We call it linear homotopy because \( H \) is a linear function of the variable \( \lambda \).

In[9]:= Manipulate[ListPlot[\lambda \text{circle} + (1 - \lambda) \text{square},
Joined -> True, AspectRatio -> 1, ImageSize -> 200], {\lambda, 0, 1}]
Out[9]=

Fig. 5.1 Homotopy between square and circle

Lines are also geometric objects. Let us consider two Bezier splines with control points,

In[13]:= pts1 = {{0, -1}, (2, 1), (4, 2), (6, 2)};
pts2 = {{2, -1}, (3, 1), (4, -1), (6, 0)};
and a homotopy between them
Now, we apply this concept to algebraic rather than geometric objects. Let us consider two univariate polynomials $p_1$ and $q_1$ of the same degree,

\[
p_1(x) = x^2 - 3 \\
q_1(x) = -x^2 - x + 1
\]

Define the linear convex function, $H$ in variables $x$ and $\lambda$, called as homotopy function as,

\[
H(x, \lambda) = (1 - \lambda)p_1(x) + \lambda q_1(x)
\]

Now, the homotopy $H$ provides us a continuous, smooth deformation from $p_1$ -which is obtained for $\lambda = 0$ by $H(x, 0)$ - to $q_1$ - which is obtained for $\lambda = 1$ by $H(x, 1)$.

Let us define these functions,

\[
p_1[x_] := x^2 - 3 \\
q_1[x_] := -x^2 - x + 1 \\
H[x_, \lambda_] := (1 - \lambda) p_1[x] + \lambda q_1[x]
\]

Now, we can plot the homotopy function with different constant $\lambda$ values, illustrating the deformation of $p_1(x)$ into $q_1(x)$.
Now let us consider two univariate polynomials \( p_2 \) and \( q_2 \) of the different degrees,

\[
p_2(x) = x^2 - 1
\]
\[
q_2(x) = \left(x^2 - \frac{1}{4}\right)(x^2 - 4)
\]

Define the linear convex homotopy again with,

\[
H(x, \lambda) = (1 - \lambda) p_2(x) + \lambda q_2(x)
\]

Similarly, the functions are

\[
p_2[x_] := x^2 - 1
\]
\[
q_2[x_] := \left(x^2 - \frac{1}{4}\right)(x^2 - 4)
\]
\[
H[x_\_, \lambda_\_] := (1 - \lambda) p_2[x] + \lambda q_2[x]
\]

and

Show[{
Plot[Table[H[x, \lambda], \{\lambda, 0, 1, 0.1\}],
{x, -2, 1.5}, AxesLabel -> {"x"}, ImageSize -> 300, PlotStyle -> Thin],
Graphics[Text["p_2", \{-0.5, -3.\}]], Graphics[Text["q_2", \{0.35, 0.8\}]],
Graphics[Text["H(x,\lambda)", \{0.3, 1.25\}]]}
]

Fig. 5.3 Deformation of the function \( H \) from \( p_1 \) to \( q_1 \) as function of \( \lambda \)

5-3 Solving nonlinear equation via homotopy

Homotopy continuation method deforms the known roots of the start system into the roots of the target system. Now, let us look at how homotopy can be used to solve a simple polynomial equation. Consider the polynomial equation of degree two,
\[ q(x) = x^2 + 8x - 9 = 0 \]

By deleting the middle term, we can get a more simple equation, which can be solved easily by inspection,
\[ p(x) := x^2 - 9 \]

This equation also has two roots and will be considered as start system for the target system. The linear homotopy can be defined as it follows
\[ H[x_\_\_\_, \_\_] := (1 - \lambda) p[x] + \lambda q[x] \]

or
\[ H[x, \lambda] // Expand \]
\[-9 + x^2 + 8x \lambda \]

Let us plot the homotopy \( H \) for the polynomials \( p(x) \) and \( q(x) \)
\[
\text{Show[}
\{
\text{Plot[Table[H[x, \lambda], \{\lambda, 0, 1, 0.1\}, \{x, -10, 10\}, AxesLabel \rightarrow \{"x"\}, PlotStyle \rightarrow \text{Thin}],}
\text{Graphics[Text["p(x)", \{8, 35\}], Graphics[Text["q(x)", \{8, 140\}]]]}
\}
\]

Fig. 5.5 Deformation of the function \( H \) from \( p(x) \) to \( q(x) \) as function of \( \lambda \)

Homotopy continuation method deforms \( p(x) = 0 \), the known roots of the start system, into \( q(x) = 0 \), the roots of the target system. Let us solve the equation \( H(x, \lambda) = 0 \) for different values of \( \lambda \). Considering \( x_0 = 3 \) one of the solutions of \( p(x) = 0 \) as initial guess value, and solving \( H(x, \lambda_1) = 0 \), where let \( \lambda_1 = 0.2 \), employing Newton-Raphson method subsequently, we get
\[ x_0 = 3; \lambda_1 = 0.2; \ x_1 = x / . \text{FindRoot[H[x, \lambda_1] \rightarrow 0, \{x, x_0\}] \}
\]
\[ 2.30483 \]

Using the result as guess value for the next solution step
\[ \lambda_2 = 0.4; \ x_2 = x / . \text{FindRoot[H[x, \lambda_2] \rightarrow 0, \{x, x_1\}] \}
\]
\[ 1.8 \]

and so on,
\[ \lambda_3 = 0.6; \ x_3 = x / . \text{FindRoot[H[x, \lambda_3] \rightarrow 0, \{x, x_2\}] \}
\]
\[ 1.44187 \]
\[ \lambda_4 = 0.8; \ x_4 = x / . \text{FindRoot[H[x, \lambda_4] \rightarrow 0, \{x, x_3\}] \}
\]
\[ 1.18634 \]
\[ \lambda_5 = 1; \ x_5 = x / . \text{FindRoot[H[x, \lambda_5] \rightarrow 0, \{x, x_4\}] \}
\]
\[ 1. \]

Let us display the transition of a root of the polynomial \( p(x) \) into a root of the polynomial \( q(x) \),
The homotopy path is the function $x = x(\lambda)$, where in every point $H(x,\lambda) = 0$.

Fig. 5.7 shows the path of homotopy transition of the root of $p(x)$ into the root of $q(x)$.

$$\lambda_0 = 0.;$$

Show[
{ListPlot[Table[(\lambda_i, x_i), \{i, 0, 5\}],
 Joined -> True, PlotStyle -> Thin, AxesLabel -> {"\lambda", "x(\lambda)"}],
 ListPlot[Table[(\lambda_i, x_i), \{i, 0, 5\}], PlotStyle -> {PointSize[0.016], RGBColor[1, 0, 0]}],
 Graphics[{Text["p(3)=0", \{0.12, 2.95\}], Text["q(1)=0", \{0.95, 1.25\}], Text["\lambda"]}
]}

5-4 Tracing homotopy path as initial value problem

Comparing homotopy solution with the traditional Newton-Raphson solution, it is clear that if $\Delta \lambda$ is small enough, the convergence may be ensured in every step.

However, one can consider this root tracing procedure as an initial value problem of an ordinary differential equation. Since $H(x, \lambda) = 0$ for every $\lambda \in [0, 1]$, therefore

$$dH(x, \lambda) = \frac{\partial H}{\partial x} dx + \frac{\partial H}{\partial \lambda} d\lambda \equiv 0 \quad \lambda \in [0, 1]$$
Then the initial value problem is
\[ H \frac{dx(\lambda)}{d\lambda} + H_x = 0 \]
with
\[ x(0) = x_0 \]
Here \( H_x \) is the Jacobian of \( H \) with respect to \( x_i \), \( i = 1, \ldots, n \), in case of \( n \) nonlinear equations with \( n \) variables.

In our single variable case, the two partial derivatives of the homotopy function are
\[ dHd\lambda = D[H[x, \lambda], \lambda] \]
\[ 8 \times \]
\[ dHdx = D[H[x, \lambda], x] \]
\[ 2 \times (1 - \lambda) + (8 + 2 \times \lambda) \]
Then the right hand side of the differential equation to be solved is
\[
deqrhs = -\frac{dHd\lambda}{dHdx} . x \rightarrow x[\lambda] \]
\[ \frac{8 \times [\lambda]}{2 \times (1 - \lambda) \times [\lambda] + \lambda \times (8 + 2 \times [\lambda])} \]
The differential equation,
\[ deq = D[x[\lambda], \lambda] \rightarrow deqrhs \]
\[ x'[\lambda] = -\frac{8 \times [\lambda]}{2 \times (1 - \lambda) \times [\lambda] + \lambda \times (8 + 2 \times [\lambda])} \]
The initial value is
\[ x0 = x_0 \]
3
The numerical solution
\[ sol = NDSolve[{deq, x[0] == x0}, \{x[\lambda]\}, \{\lambda, 0, 1\}]; \]
The trajectory is the homotopy path.
\[ Plot[x[\lambda] /. sol, \{\lambda, 0, 1\}, PlotStyle \rightarrow Thin, \]
\[ AxesLabel \rightarrow \{"\lambda", "x(\lambda)"\}, Epilog \rightarrow\{PointSize[0.016], Blue, Point\[\{0, x0\}], \]
\[ PointSize[0.016], Red, Point\[\{1, First[(x[\lambda] /. sol) /. \lambda \rightarrow 1]\}\}]] \]

The value of the corresponding root of \( q(x) \) is \( x(\lambda) \) at \( \lambda = 1 \).
1. \textbf{5-5 Types of linear homotopy}

\textit{5-5-1 General linear homotopy}

As we have seen, the start system can be constructed intuitively, reducing the original system (target system) to a more simple system (start system), which roots can be easily computed. In order to get all of the roots of the target system, the start system should have so many roots as many the target system has.

The start system can be constructed in different ways, however there are two typical techniques for generating the start systems, which are usually employed.

\textit{5-5-2 Fixed point homotopy}

The start system can be considered as

\[ p(x) = x - x_0 \]

where \( x_0 \) is a guess value for the root of the target system \( q(x) = 0 \).

In that case the homotopy function is

\[ H(x, \lambda) = (1 - \lambda)(x - x_0) + \lambda q(x) \]

\textit{5-5-3 Newton homotopy}

An other construction for the start system, when we consider \( p(x) \) as

\[ p(x) = q(x) - q(x_0) \]

in that case the homotopy function is

\[ H(x, \lambda) = (1 - \lambda)(q(x) - q(x_0)) + \lambda q(x) \]

or

\[ H(x, \lambda) = q(x) - (1 - \lambda)q(x_0) \]

\textit{5-5-4 Affine homotopy}

This type of homotopy suggested by Jalali and Seader (2000) requires the first derivate of the target function,

\[ H(x, \lambda) = (1 - \lambda)q'(x_0)(x - x_0) + \lambda q(x) \]

Undoubtedly, this can effectively employed when the derivate can given in analytical form. In case of polynomials it is the case. It goes without saying that for system of equations the Jacobian should be used.

\textit{5-5-5 Mixed homotopy}

Rahimian et al. (2011) concerned with the use of homotopy function,

\[ H(x, \lambda) = (1 - \lambda)(q(x) - q(x_0) + (x - x_0)) + \lambda q(x) \]

to track the approximate solution. Here the start system is a linear combination of the fixed point and the Newton homotopy.

There are other methods to construct start system for linear homotopy. In a section, following later, we shall see how one can define start system for polynomial systems automatically.

It is known that local methods, like Newton-Raphson method, require initial guess of a root, which is close to the
intended root for the particular application. Global methods, like homotopy continuation method can find solution from a start guess, which is far from the solution. Now, we shall illustrate some situations when Newton-Raphson method fails, but homotopy method can be successful. We shall consider two usual problems with the Newton-Raphson method, namely when the Jacobi matrix is singular or when the convergence is too slow.

5-6 Newton-Raphson vs. homotopy method

5-6-1 Singular Jacobi matrix

Let us consider the following univariate polynomial of degree three,

\[ q(x) = 27x^3 + 108x^2 + 108x + 40 \]

This polynomial has one real and two conjugate complex roots.

\[ q[x_] := 27x^3 + 108x^2 + 108x + 40 \]

Plot \[ q(x), \{x, -3, 1\} \], AxesLabel \[ \{"x", "q(x)"\} \], PlotStyle \[ \rightarrow \text{Thin} \], ImageSize \[ \rightarrow 350 \]

![Fig. 5.9 The real root of the polynomial](image)

Let us try to find the real root using Newton-Raphson method, with an initial guess value \( x_0 = -2 \),

\[ \text{FindRoot}[q[x], \{x, -2\}] \]

FindRoot::sing: Encountered a singular Jacobian at the point \( \{x\} = \{-2\} \). Try perturbing the initial point(s). \( \rightarrow \)

\[ \{x \rightarrow -2.\} \]

This method also fails when \( x_0 = -1.5 \)

\[ \text{FindRoot}[q[x], \{x, -1.5\}] \]

FindRoot::lstol:

The line search decreased the step size to within tolerance specified by AccuracyGoal and PrecisionGoal but was unable to find a sufficient decrease in the merit function. You may need more than MachinePrecision digits of working precision to meet these tolerances. \( \rightarrow \)

\[ \{x \rightarrow -0.666667\} \]

Now we use homotopy method. Let us employ fixed-point homotopy with \( x_0 = -2 \),

Fig. 5.10 shows the plot of fixed-point homotopy curves for target system \( q(x) \) in case of \( x_0 = -2 \), \( x_0 = -2 \);

\[ H[x_, \lambda_] := (1 - \lambda)(x - x_0) + \lambda q[x] \]

Show[{Plot[Table[H[x, \lambda], \{\lambda, 0, 1, 0.1\}], \{x, -3, -2\}, PlotStyle \[ \rightarrow \text{Thin} \],

AxesLabel \[ \rightarrow \{"x", "p(x),q(x)"\} \], PlotRange \rightarrow \{-10, 40\}, ImageSize \rightarrow 350\},

Graphics[{Text["p(x)=x-x_0", \{-2.15, 1.5\}], Text["q(x)\(\), \{-2.45, 31\}]}\]}]
Let us generate the fixed-point homotopy path for \( q(x) = 0 \) in case of \( x_0 = -2 \),

\[
\text{rhs} = \frac{D[H[x, \lambda], \lambda]}{D[H[x, \lambda], x]} / x \rightarrow x[\lambda] ;
\]

\[
\text{sol} = \text{NDSolve}\{D[H[x[\lambda], \lambda] = \text{rhs}, x[0] = x0], \{x[\lambda]\}, \{\lambda, 0, 1\} ;
\]

\[
\text{Plot}[x[\lambda] /. \text{sol}, (\lambda, 0, 1), \text{PlotStyle} \rightarrow \text{Thin},
\]

\[
\text{PlotRange} \rightarrow (-1.9, -3), \text{AxesLabel} \rightarrow \{"\lambda", "x(\lambda) "\},
\]

\[
\text{Epilog} \rightarrow \{\text{PointSize}[0.016], \text{Blue}, \text{Point}[\{0, x0\}], \text{PointSize}[0.016],
\]

\[
\text{Red}, \text{Point}[\{1, \text{First}[\{x[\lambda] / . \text{sol} / . \lambda \rightarrow 1\}])], \text{ImageSize} \rightarrow 350\]

A value of the real root is

\[
\text{First}[x[\lambda] / . \text{sol} / . \lambda \rightarrow 1] = -2.73587
\]

This fixed-point homotopy also works with \( x_0 = -1.5 \).

5-6-2 Slow convergence

Let us consider now a two-dimensional problem. The target system is

\[
f_1(x, y) = x^2 - xy - 1
\]

\[
f_2(x, y) = \sqrt{x + 1} + y
\]

Fig. 5.12 shows the contour plot of the system of equations in case of \( f_1 = 0 \) and \( f_2 = 0 \).
\( f_1[x_, y_] := x^2 - x \cdot y - 1 \)
\( f_2[x_, y_] := \sqrt{x + 1} + y \)

Show[
{ContourPlot[{f1[x, y] == 0, f2[x, y] == 0}, {x, -1.1, 2}, {y, -2, 1}, ContourStyle \to Thin,
FrameLabel \to \{"x", "y"\}, ImageSize \to 350], Graphics[{Text["f_1(x, y) = 0", \{1.43, 0\}],
Text["f_2(x, y) = 0", \{-0.5, -1\}], Text["f_1(x, y) = 0", \{-0.45, 0.3\}]}]}
]

We try Newton-Raphson method with \( x_0 = -0.1 \) and \( y_0 = 0.5 \).
\[ x_0 = -0.1; \quad y_0 = 0.5; \]
FindRoot[{f1[x, y], f2[x, y]}, \{x, x0\}, \{y, y0\}]

FindRoot::cvmit: Failed to converge to the requested accuracy or precision within 100 iterations. >> 
\{x \to -1. - 0.0000512881 \text{i}, \ y \to 4.30262 \times 10^{-7} - 0.000102576 \text{i} \}

Now let us employ Newton homotopy,
\[ H[x_, y_, \lambda_] := Flatten[\{-1 + \lambda \ x, \ f_1[x, y] + \ f_2[x, y] \}] \]
\[ H[x, y, \lambda] \]
\[ \{-1 + x^2 - x \cdot y - 0.94 (-1 + \lambda), \ \sqrt{1 + x} + 1.44868 (-1 + \lambda) \} \]

Let us generate the contour plot of the start system in case of Newton homotopy with \( x_0 = -0.1 \) and \( y_0 = 0.5 \).
Show[{ContourPlot[H[x, y, 0][[1]] == 0, \{x, -1, 2\}, \{y, -2, 1\},
ContourStyle \to Thin, FrameLabel \to \{"x", "y"\}, ImageSize \to 350],
Graphics[{Text["H(x, y, 0) = 0", \{0.75, -1\}], ContourPlot[H[x, y, 0][[2]] == 0,
\{x, -1, 2\}, \{y, -2, 1\}, ContourStyle \to Thin, ImageSize \to 350]}]
]
Let us define a function for the Jacobi computation as it follows,

\[ \text{Jacobi}[F_\lambda, X_] := \text{Outer}[D, F, X] \]

Then the total Jacobian, \( (\frac{\partial H}{\partial x} \frac{\partial H}{\partial y} \frac{\partial H}{\partial \lambda}) \)

\[ J_{xy} \lambda = \text{Jacobi}[H[x, y, \lambda], \{x, y, \lambda\}] ; \text{MatrixForm}[J_{xy} \lambda] \]

\[
\begin{pmatrix}
2x - y & -x & -0.94 \\
\frac{1}{2\sqrt{1+x}} & 1 & 1.44868
\end{pmatrix}
\]

The Jacobian respecting to \( x \) and \( y \)

\[ J_{xy} = \text{Take}[J_{xy} \lambda, \{1, 2\}, \{1, 2\}] ; \text{MatrixForm}[J_{xy}] \]

\[
\begin{pmatrix}
2x - y & -x \\
\frac{1}{2\sqrt{1+x}} & 1
\end{pmatrix}
\]

Now, in case of system of two variables, \( X = [x, y] \), we should solve a linear system in order to get explicit form of the differential equation system,

\[ H_X \frac{dX(\lambda)}{d\lambda} + H_\lambda = 0 \]

namely

\[
\frac{dX(\lambda)}{d\lambda} = H_X^{-1} H_\lambda
\]

The right hand side of the linear system

\[ b = -\text{Take}[J_{xy} \lambda, \{1, 2\}, \{2+1\}] ; \text{MatrixForm}[b] \]

\[
\begin{pmatrix}
0.94 \\
-1.44868
\end{pmatrix}
\]

Solving the linear system, we get
\[
\text{rhs} = \text{LinearSolve}[\text{Jxy}, b] \quad \text{// Simplify} /\text{. Map}[\text{\[Pi]} \rightarrow \pi[\lambda] \&, \{x, y\}] \quad \text{// Flatten}

\left\{
\begin{aligned}
&\left(\sqrt{1. + x[\lambda]} \right) ^{-1} \left(0.724342 x[\lambda]^2 + x[\lambda] \left(0.47 + 0.362171 y[\lambda]\right) - 0.235 y[\lambda]\right) \\
&\left(0.25 x[\lambda] + 1. x[\lambda] \sqrt{1. + x[\lambda]} - 0.5 \sqrt{1. + x[\lambda]} y[\lambda]\right), \\
&\left(-\left(2.89737 x[\lambda] + \frac{0.47}{\sqrt{1. + x[\lambda]}} - 1.44868 y[\lambda]\right) / \left(2. x[\lambda] + \frac{0.5 x[\lambda]}{\sqrt{1. + x[\lambda]}} - 1. y[\lambda]\right)\right)
\end{aligned}
\right\}
\]

Then the left hand side of the differential equation system is,

\[
\text{lhs} = \text{Map}[\text{D}[\text{\[Pi]}[\lambda], \{\lambda, 1\}] \&, \{x, y\}]
\]

\{x'[\lambda], y'[\lambda]\}

Therefore the differential equation system,

\[
\text{deqs} = \text{MapThread}[\text{\[Pi]}1 = \text{\[Pi]}2 \&, \{\text{lhs}, \text{SetPrecision}[\text{rhs}, 20]\}]
\]

\[
\begin{aligned}
x'[\lambda] &= \left(\sqrt{1.0000000000000000000 x[\lambda]} - 0.5000000000000000000 x[\lambda]\right) \\
&\quad \left(0.25000000000000000000 x[\lambda] + \\
&\quad 1.000000000000000000000 x[\lambda] \sqrt{1.0000000000000000000 x[\lambda]} - \\
&\quad 0.50000000000000000000 x[\lambda] \sqrt{1.0000000000000000000 x[\lambda]}\right) \\
y'[\lambda] &= -\left(2.000000000000000000000 x[\lambda] - 0.5000000000000000000000 x[\lambda]\right) / \\
&\quad \left(\sqrt{1.00000000000000000000 x[\lambda]} - 1.00000000000000000000 x[\lambda]\right)
\end{aligned}
\]

Here we used an extra precision because this system is stiff. The initial values are also given in rational form representing infinite precision. The numerical solution,

\[
\text{sol} = \text{NDSolve}[\text{Join[deqs}, \{x[0] = -1/10, y[0] = 1/2\}], \{x[\lambda], y[\lambda]\}, \{\lambda, 0, 1\}, \text{WorkingPrecision} \rightarrow 20]
\]

The plot of the homotopy solution of the system in case of Newton homotopy with \(x_0 = -0.1\) and \(y_0 = 0.5.\)

\[
\text{Show} \left[\text{Plot}\left[\{\{x[\lambda], y[\lambda]\} /\text{. sol, } \{\lambda, 0, 1\}, \text{PlotStyle} \rightarrow \text{Thin, FrameLabel} \rightarrow \{"\lambda", "x(\lambda), y(\lambda)\"\}, \text{Frame} \rightarrow \text{True, Epilog} \rightarrow \{\text{PointSize}[0.016], \text{Blue, Point}[\{0, x[0]\}], \text{PointSize}[0.016], \text{Red, Point[[1, First[\{x[\lambda] /\text{. sol} /\lambda \rightarrow 1\}], \{1, First[\{y[\lambda] /\text{. sol} /\lambda \rightarrow 1\}]\}]\}], \text{Graphics}[\text{Text}["x(\lambda)", \{0.6, -0.6\}], \text{Graphics}[\text{Text}["y(\lambda)", \{0.4, 0.4\}]\}\]\]
\]
The solution of the system,
\[ \{xs, ys\} = Flatten[((x[\lambda], y[\lambda]) /. sol) /. \lambda \to 1] \]
\[ \{-0.9999999999999999002, -6.5296973861 \times 10^{-10}\} \]

We can round these values,
\[ \{xs, ys\} = \{xs, ys\} // Round \]
\[ \{-1, 0\} \]

5-7 Regularization of the homotopy function

Sometimes this form of homotopy methods may also fail, because of singularity resulting diverging path. Let us consider the following system,

\[ f_1(x, y) = x^2 + y - 3 \]
\[ f_2(x, y) = \frac{1}{8} y^2 - 1 \]

\[ f1[x_, y_] := x^2 + y - 3 \]
\[ f2[x_, y_] := x + \frac{1}{8} y^2 - 1 \]

Show[{ContourPlot[{f1[x, y] == 0, f2[x, y] == 0}, {x, -10, 10}, {y, -10, 10}, ContourStyle -> Thin, FrameLabel -> {"x", "y"}], Graphics[{Text["f2(x, y) = 0", {-8, 6.5}], Text["f1(x, y) = 0", {0, -9}]}]}]

First, let us try to solve the problem using Newton-Raphson method, starting with \( x_0 = -1 \) and \( y_0 = -1 \).

\[ \text{FindRoot[\{f1[x, y], f2[x, y]\}, \{x, -1\}, \{y, -1\}] \]
This result is not correct,
\[ \{f1[x, y], f2[x, y]\} \%
\{0.729457, 0.769873\}
\]

or with \(x_0 = 1\) and \(y_0 = -1\),
\[
\text{FindRoot}[[f1[x, y], f2[x, y]], \{x, 1\}, \{y, -1\}]
\]

\text{FindRoot::lstol: }
The line search decreased the step size to within tolerance specified by \text{AccuracyGoal} and \text{PrecisionGoal} but was unable to find a sufficient decrease in the merit function. You may need more than \text{MachinePrecision} digits of working precision to meet these tolerances.

\[
\{x \rightarrow 1.27083, y \rightarrow 1.57379\}
\]

again, this is not a solution
\[
\{f1[x, y], f2[x, y]\} \%
\{0.188785, 0.580427\}
\]

These mean that none of them is a correct solution, Newton - Raphson method fails.

Let us try to solve the problem with homotopy method. We consider the following obvious functions as start system of \([f_1, f_2]\) by deleting the low order terms,
\[
g1[x, \_] := x^2 - 1
\]
\[
g2[x, \_] := y^2 - 1
\]

\textit{Remark:} The degree of the start and the target system should be the same.

\[
\text{Solve}[[g1[x, y] = 0, g2[x, y] = 0], \{x, y\}]
\]

\[
\{\{x \rightarrow -1, y \rightarrow -1\}, \{x \rightarrow 1, y \rightarrow -1\}, \{x \rightarrow -1, y \rightarrow 1\}, \{x \rightarrow 1, y \rightarrow 1\}\}
\]

The zeros of this system are \((1, 1), (-1, 1), (-1,-1)\) and \((1,-1)\). Let us consider \(x_0 = 1; y_0 = -1\).

The homotopy function is
\[
H[x, y, \_] := \text{Flatten}[[1 - \lambda \left( g1[x, y] \right) + \lambda \left( f1[x, y] \right)]]
\]
\[
H[x, y, \lambda] = \left\{ (-1 + x^2) \ (1 - \lambda) + (-3 + x^2 + y) \ \lambda, \ (-1 + y^2) \ (1 - \lambda) + \left\{ -1 + x + \frac{y^2}{\delta} \right\} \lambda \right\}
\]

The total Jacobian,
\[
Jx\lambda = \text{Jacobi}[H[x, y, \lambda], \{x, y, \lambda\}]; \text{MatrixForm}[Jx\lambda]
\]
\[
\left( \begin{array}{c}
2 \times (1 - \lambda) + 2 \times \lambda & -2 + \frac{y}{\delta} \\
\frac{\lambda}{2} & 1 - \lambda + \frac{x^2}{\delta}
\end{array} \right)
\]

The Jacobian respecting to \(x\) and \(y\)
\[
Jx = \text{Take}[Jx\lambda, \{1, 2\}, \{1, 2\}]; \text{MatrixForm}[Jx]
\]
\[
\left( \begin{array}{c}
2 \times (1 - \lambda) + 2 \times \lambda & -2 + \frac{y}{\delta} \\
\frac{\lambda}{2} & 1 - \lambda + \frac{x^2}{\delta}
\end{array} \right)
\]

Now, we set up the differential equation system for tracing homotopy paths. Explicit form of the differential equation system should be expressed.

The right hand side of the linear system
Solving the linear system, we get the right hand side,
\[
\text{rhs = (LinearSolve[Jx, b] // Simplify) / Map[H \rightarrow H[\lambda] & \rightarrow \{x, y\}] // Flatten}
\]
\[
\{(1 - 8 \lambda x[\lambda] + 4 (-8 + 7 \lambda) y[\lambda] + (16 - 7 \lambda) y[\lambda]^2) / (4 (2 \lambda^2 + (8 + 7 \lambda) x[\lambda] y[\lambda])),
8 x[\lambda]^2 - 4 \lambda (-2 + y[\lambda]) - 7 x[\lambda] y[\lambda]^2) / (2 (2 \lambda^2 + (8 + 7 \lambda) x[\lambda] y[\lambda]))\}
\]
The left hand side is,
\[
\text{lhs = Map[D[H[\lambda], \lambda], \{x, y\}}
\]
\[
\{x'[\lambda], \ y'[\lambda] \}
\]
Then the differential equation system can be written as,
\[
\text{deqs = MapThread[\#1 \rightarrow \#2 \&, \{lhs, rhs\}]}
\]
\[
\{x'[\lambda] = (-8 \lambda x[\lambda] + 4 (-8 + 7 \lambda) y[\lambda] + (16 - 7 \lambda) y[\lambda]^2) / (4 (2 \lambda^2 + (8 + 7 \lambda) x[\lambda] y[\lambda])),
y'[\lambda] = (8 x[\lambda]^2 - 4 \lambda (-2 + y[\lambda]) - 7 x[\lambda] y[\lambda]^2) / (2 (2 \lambda^2 + (8 + 7 \lambda) x[\lambda] y[\lambda]))\}
\]
The numerical solution of the system with the initial values \{x0, y0\},
\[
\text{sol = NDSolve[Join[deqs, \{x[0] = x0, y[0] = y0\}],}
\]
\[
\{x[\lambda], y[\lambda]\}, \{\lambda, 0, 1\}, \text{Method \rightarrow "ExplicitRungeKutta"}];
\]
\text{NDSolve::nstf: At \lambda \approx 0.663818254320564, system appears to be stiff. Methods Automatic, BDF, or StiffnessSwitching may be more appropriate. \Rightarrow}

or
\[
\text{sol = NDSolve[Join[deqs, \{x[0] = x0, y[0] = y0\}],}
\]
\[
\{x[\lambda], y[\lambda]\}, \{\lambda, 0, 1\}, \text{Method \rightarrow "StiffnessSwitching"}];
\]
\text{NDSolve::ndsz: At \lambda \approx 0.6638182463118296, step size is effectively zero; singularity or stiff system suspected. \Rightarrow}

The integration has failed because singularity of the Jacobian matrix of \( H(x, y, \lambda) \) occurred.

In order to avoid singularity in real field, we consider a modified complex homotopy function,
\[
H(x, \lambda) = y(1 - \lambda) p_1(x) + \lambda q_1(x)
\]
where \( \gamma \) is a complex number. For almost all choices of a complex constant \( \gamma \), all solution paths defined by the homotopy above are regular, i.e.: for all \( \lambda \in [0, 1] \), the Jacobian matrix of \( H(x, \lambda) \) is regular and no path diverges.

Remark
Alternatively, Kalaba and Tesfatsion (1991) proposed that the \( \lambda \) parameter should move from \( 0 + 0i \) to \( 1 + 0i \) along a spider-web grid centered at \( 1 + 0i \) in the complex plain. The actual path through the grid is determined adaptively, step by step in accordance with two objectives: short path length, and avoidance of singular points.
Let us now consider, 

\[ y = 1 + i; \]

Then the homotopy function,

\[
H[x, y, \lambda] := \text{Flatten} \left[ (1 - \lambda) \left( \begin{array}{c} g1[x, y] \\ g2[x, y] \end{array} \right) + \lambda \left( \begin{array}{c} f1[x, y] \\ f2[x, y] \end{array} \right) \right]
\]

\[
H[x, y, \lambda] = \left\{ (1 + i) (-1 + x^2) (1 - \lambda) + (-3 + x^2 + y) \lambda, (1 + i) (-1 + y^2) (1 - \lambda) + \left(-1 + x + \frac{y^2}{8}\right) \lambda \right\}
\]

The total Jacobian,

\[
Jx = \text{Take}[Jx, \lambda] \left( \begin{array}{c} (2 + 2 i) x (1 - \lambda) + 2 x \lambda \\ \lambda \end{array} \right)
\]

\[
J \lambda = \text{Jacobi}[H[x, y, \lambda], (x, y, \lambda)] \text{; MatrixForm}[Jx \lambda]
\]

\[
Jx = \text{Take}[Jx, \lambda] \left( \begin{array}{c} (2 + 2 i) x (1 - \lambda) + 2 x \lambda \\ \lambda \end{array} \right)
\]

\[
J \lambda = \text{Take}[Jx, \lambda] \left( \begin{array}{c} (2 + 2 i) x (1 - \lambda) + 2 x \lambda \\ \lambda \end{array} \right)
\]

The right hand side of the linear system

\[
b = -\text{Take}[Jx, \lambda] \left( \begin{array}{c} 3 - x^2 + (1 + i) (-1 + x^2) - y \\ 1 - x - \frac{y}{i} + (1 + i) (-1 + y^2) \end{array} \right)
\]

Solving the linear system,

\[
\text{rhs} = \left( \text{LinearSolve}[Jx, b] \right) \left( \begin{array}{c} \text{Simplify} / \text{Map} \left[ \text{Re}[x \lambda], (x, y) \right] \right) \left( \begin{array}{c} \text{Flatten} \left[ \left( \begin{array}{c} (8 \lambda (1 - i x [\lambda]) - 2 ((-8 - 8 i) + (7 + 8 i) \lambda) \left( -1 - 2 i \right) + x [\lambda] \right) y [\lambda] + ((-16 + 16 i) + (8 - 7 i) \lambda) y [\lambda]^2 \right) / (4 \left(2 i \lambda^2 + (16 - (23 + 9 i) \lambda + (7 + 8 i) x [\lambda] y [\lambda]) \right)), \\
\left\{ (1 + 2 i + \lambda) x [\lambda]^2 + 4 \lambda ((1 + 2 i) - \lambda y [\lambda]) - ((-1 + 3 x [\lambda] + x [\lambda] \left( -8 + (7 + 8 i) y [\lambda]^2) / (2 \left(2 i \lambda^2 + (16 - (23 + 9 i) \lambda + (7 + 8 i) x [\lambda] y [\lambda]) \right) \right) \right) \right\}
\right.
\]

Then the differential equation system is,
\begin{align*}
\text{lhs} &= \text{Map[D[H[\lambda], (\lambda, 1)] &, \{x, y\}]}
\text{x'[\lambda], y'[\lambda]}\\
\text{deqs} &= \text{MapThread[\(H1 := H2 \&\), \{lhs, rhs\}]}
\{x'[\lambda] = \left(3 \lambda - x[\lambda] \right) - \\
&\quad \left(2 \left(-1 + x[\lambda] + \lambda \right) \left(-1 + x[\lambda] \right) \right) y[\lambda] + \\
&\quad \left((-1 + x[\lambda]) \left(16 + \lambda \lambda \right) \right) + \\
&\quad \lambda \left(-1 + x[\lambda] \right) \lambda \right) \right) + \\
&\quad \lambda \left(16 + \lambda \left(7 + 8 \lambda \right) \right) y[\lambda] \right) \right) / \\
&\quad \lambda \left(16 + \lambda \left(7 + 8 \lambda \right) \right) y[\lambda] \right) \right) / \\
&\quad \lambda \left(16 + \lambda \left(7 + 8 \lambda \right) \right) y[\lambda] \right) \right) \}
\end{align*}

The numerical solution
\begin{align*}
\text{sol} &= \text{NDSolve[Join[deqs, \{x[0] = x0, y[0] = y0\}], \{x[\lambda], y[\lambda]\}, \{\lambda, 0, 1\}]};
\end{align*}

The solution of the polynomial system,
\begin{align*}
\{(x[\lambda], y[\lambda]) / . \text{sol} / . \lambda \to 1\}
\end{align*}

Now we have got a complex root,
\begin{align*}
\text{Show[\{ParametricPlot[\{Re[y[\lambda]], Im[y[\lambda]]\} / . \text{sol}, \{\lambda, 0, 1\}],}
\text{PlotStyle} \to \text{Thin}, \text{PlotRange} \to \text{All}, \text{FrameLabel} \to \{"Re", "Im"\}, \text{Frame} \to \text{True},
\text{Epilog} \to \{\text{PointSize}[0.02], \text{Blue}, \text{Point[\{Re[y0], Im[y0]\}]}, \text{PointSize}[0.02], \text{Red},
\text{Point[\{First[\{Re[y[\lambda]] / . \text{sol} / . \lambda \to 1\}], First[\{Im[y[\lambda]] / . \text{sol} / . \lambda \to 1\}]},
\text{PointSize}[0.02], \text{Blue}, \text{Point[\{Re[x0], Im[x0]\}]}, \text{PointSize}[0.02], \text{Red},
\text{Point[\{First[\{Re[x[\lambda]] / . \text{sol} / . \lambda \to 1\}], First[\{Im[x[\lambda]] / . \text{sol} / . \lambda \to 1\}]}}
\end{align*}

\begin{align*}
\text{ParametricPlot[\{Re[x[\lambda]], Im[x[\lambda]]\} / . \text{sol}, \{\lambda, 0, 1\}],}
\text{PlotStyle} \to \text{Thin}, \text{PlotRange} \to \text{All},
\text{Graphics[\{Text["x[\lambda]", \{1.5, 0.5\}], Graphics[Text["y[\lambda]", \{0.5, -2\}]\}]
\end{align*}

Basically, our original system, the target system has four solutions. Employing built-in numerical Groebner basis
\textbf{NSolve}\([\{f1[x, y] = 0, f2[x, y] = 0\}, \{x, y\}]\)

\{\(x \rightarrow 1.5247 + 0.797778 \text{i}, y \rightarrow 1.31173 - 2.43275 \text{i}\)\},

\{\(x \rightarrow 1.5247 - 0.797778 \text{i}, y \rightarrow 1.31173 + 2.43275 \text{i}\)\},

\{\(x \rightarrow -0.115088, y \rightarrow 2.98675\), \(x \rightarrow -2.93432, y \rightarrow -5.61022\)\}

The other roots can be also computed using different start values, namely

\[x_0 = 1; \ y_0 = 1;\]

\(\text{sol} = \text{NSolve}[\text{Join}[\text{deqs}, \{x[0] = x_0, y[0] = y_0\}], \{x[\lambda], y[\lambda]\}, \{\lambda, 0, 1\}]\);

\(\{(x[\lambda], y[\lambda]) \/. \text{sol} \}/. \lambda \rightarrow 1\)

\{\(-0.115088 + 1.11099 \times 10^{-6} \text{i}, 2.98675 - 5.60973 \times 10^{-7} \text{i}\}\}

These complex 'tails' can be eliminated by using computation with high precision.

\(\text{sol} = \text{NSolve}[\text{Join}[\text{deqs}, \{x[0] = x_0, y[0] = y_0\}], \{x[\lambda], y[\lambda]\}, \{\lambda, 0, 1\}]\);

\(\{(x[\lambda], y[\lambda]) \/. \text{sol} \}/. \lambda \rightarrow 1\)

\{\(-0.11508799467984834341983143 + 1.30862908320 \times 10^{-18} \text{i}, 2.9867547534805717697308331834 - 4.13551676282 \times 10^{-19} \text{i}\}\}

\(\text{Chop}[\%] \// \text{N}\)

\{\(-0.115088, 2.98675\}\}

The other real root can be computed similarly,

\[x_0 = -1; \ y_0 = 1;\]

\(\text{sol} = \text{NSolve}[\text{Join}[\text{deqs}, \{x[0] = x_0, y[0] = y_0\}], \{x[\lambda], y[\lambda]\}, \{\lambda, 0, 1\}]\);

\(\{(x[\lambda], y[\lambda]) \/. \text{sol} \}/. \lambda \rightarrow 1\)

\{\(-2.93432 - 1.62631 \times 10^{-7} \text{i}, -5.61022 - 8.37213 \times 10^{-7} \text{i}\}\}

Now, again complex 'tails' can be eliminated.

\(\text{sol} = \text{NSolve}[\text{Join}[\text{deqs}, \{x[0] = x_0, y[0] = y_0\}], \{x[\lambda], y[\lambda]\}, \{\lambda, 0, 1\}]\);

\(\{(x[\lambda], y[\lambda]) \/. \text{sol} \}/. \lambda \rightarrow 1\)

\{\(-2.93431716517985510199823821617 - 3.4616114893 \times 10^{-19} \text{i}, -5.6102172258691410513129891206 - 1.6563990761 \times 10^{-18} \text{i}\}\}

\(\text{Chop}[\%] \// \text{N}\)

\{-2.93432, -5.61022\}

\section*{5-8 Start system for polynomial systems}

How we can find the proper start system, which will provide all of the solutions of the target system automatically? This problem can be solved if the nonlinear system is specially a system of polynomial equations.

Let us consider the case when we are looking for the homotopy solution of \(f(x) = 0\), where \(f(x)\) is a polynomial system, \(f(x): \mathbb{R}^n \rightarrow \mathbb{R}^n\). To get all of the solutions, one should find out a proper polynomial system, as start
system, $g(x) = 0$, where $g(x) : \mathbb{R}^2 \to \mathbb{R}^2$ with known or easily computable solutions.

An appropriate start system can be generated in the following way,

Let $f_i(x_1, \ldots, x_n)$, $i = 1, \ldots, n$ be a system of $n$ polynomials. We are interested in the common zeros of the system, namely $f = (f_1(x), \ldots, f_n(x)) = 0$.

Let $d_i$ denote the degree of the $j$th polynomial — that is the degree of the highest order monomial in the equation. Then such a starting system is,

$$g_j(x) = e^{i\theta_j} \left( x_j^d - (e^{i\phi_j})^d \right) = 0, \quad j = 1, \ldots, n$$

where $\phi_j$ and $\theta_j$ are random real numbers in the interval $[0, 2\pi]$. The equation above has the obvious particular solution $x_j = e^{i\theta_j}$ and the complete set of the starting solutions for $j = 1, \ldots, n$ is given by

$$e^{i(\theta_j + \frac{2\pi k}{d_j})}, \quad k = 0, 1, \ldots, d_j - 1$$

**Bezout's theorem** states that the number of isolated roots of such a system is bounded by the total degree of the system,

$$\prod_{i=1}^n d_i = d_1d_2 \ldots d_n.$$

Let us consider the following system

$$\begin{align*}
f_1[x, y] :&= x^2 + y^2 - 1 \\
f_2[x, y] :&= x^3 + y^3 - 1
\end{align*}$$

The degrees of the polynomials are

$$d_1 = 2; \quad d_2 = 3;$$

Indeed, this system has the following six roots ($d_1d_2 = 2 \times 3 = 6$), as it is expected. Indeed using built-in **Global Numerical Solver**,

```math
\text{NSolve}[[f1[x, y] == 0, f2[x, y] == 0], \{x, y\}]
```

$\{(x \to -1. + 0.707107 \hat{i}, y \to -1. - 0.707107 \hat{i}), (x \to -1. - 0.707107 \hat{i}, y \to -1. + 0.707107 \hat{i}), (x \to 1., y \to 0.), (x \to 1. + 0.707107 \hat{i}, y \to 1.), (x \to 0., y \to 1.)\}$

```math
\text{Length}[%]
```

6

Now, we compute the start system. We generate random real numbers in the interval $[0, 2\pi]$ as it follows

$$\begin{align*}
\phi_1 &= \text{Random}[	ext{Real}, \{0, 2\pi\}] \\
3.09159 \\
\phi_2 &= \text{Random}[	ext{Real}, \{0, 2\pi\}] \\
5.03082 \\
\theta_1 &= \text{Random}[	ext{Real}, \{0, 2\pi\}] \\
5.40531 \\
\theta_2 &= \text{Random}[	ext{Real}, \{0, 2\pi\}] \\
3.51931
\end{align*}$$

The start system is,

$$\begin{align*}
g_1[x, y] := e^{i\phi_1} (x^{d_1} - (e^{i\theta_1})^{d_1}) \\
g_2[x, y] := e^{i\phi_2} (y^{d_2} - (e^{i\theta_2})^{d_2}) \\
g_1[x, y] \\
(-0.99875 + 0.0499858 \hat{i}) (0.183911 + 0.982943 \hat{i}) + x^2)
\end{align*}$$
\( g_2[x, y] (0.313078 - 0.949727 \dot{x}) ((0.423813 + 0.90575 \dot{y}) + y^3) \)

Then the complete set of the starting solutions,

\( \text{Xi} = \text{Table}[\text{Exp}[\dot{y} + 2 \pi \dot{k} / d1], \{k, 0, d1 - 1\}] \)

\( \{0.638784 - 0.769386 \dot{x}, -0.638784 + 0.769386 \dot{y}\} \)

and

\( \text{Yi} = \text{Table}[\text{Exp}[\dot{y} + 2 \pi \dot{k} / d2], \{k, 0, d2 - 1\}] \)

\( \{-0.92951 - 0.368797 \dot{x}, 0.784143 - 0.62058 \dot{y}, 0.145367 + 0.989378 \dot{y}\} \)

We need all of the combination of the initial values \([X_i, Y_j], i = 1, 2, j = 1, 2\)

\( \text{X0} = \text{Tuples}[[\text{Xi}, \text{Yi}]] \)

\( \{0.638784 - 0.769386 \dot{x}, -0.92951 - 0.368797 \dot{y}\}, \)

\( \{0.638784 - 0.769386 \dot{x}, 0.784143 - 0.62058 \dot{y}\}, \)

\( \{0.638784 - 0.769386 \dot{x}, 0.145367 + 0.989378 \dot{y}\}, \)

\( \{-0.638784 + 0.769386 \dot{x}, -0.92951 - 0.368797 \dot{y}\}, \)

\( \{-0.638784 + 0.769386 \dot{x}, 0.784143 - 0.62058 \dot{y}\}, \)

\( \{-0.638784 + 0.769386 \dot{x}, 0.145367 + 0.989378 \dot{y}\} \)

These values are satisfy the start system,

\( \text{Map}[g1[\#[[1]], \#[[2]]], \text{X0}] \)

\( \{0. + 0. \dot{x}, 0. + 0. \dot{y}, 0. + 0. \dot{y}, 1.99596 \times 10^{-16} + 1.01172 \times 10^{-16} \dot{y}, 1.99596 \times 10^{-16} + 1.01172 \times 10^{-16} \dot{y}, 1.99596 \times 10^{-16} + 1.01172 \times 10^{-16} \dot{y}\} \)

\( \text{Chop}[] \)

\( \{0, 0, 0, 0, 0, 0\} \)

\( \text{Map}[g2[\#[[1]], \#[[2]]], \text{X0}] \)

\( \{0. + 0. \dot{x}, 2.66515 \times 10^{-16} + 9.646 \times 10^{-16} \dot{y}, 5.3303 \times 10^{-17} + 1.9292 \times 10^{-16} \dot{y}, 0. + 0. \dot{x}, 2.66515 \times 10^{-16} + 9.646 \times 10^{-16} \dot{y}, 5.3303 \times 10^{-17} + 1.9292 \times 10^{-16} \dot{y}\} \)

\( \text{Chop}[] \)

\( \{0, 0, 0, 0, 0, 0\} \)

So we have six initial values for solving our homotopy equations. These initial values will provide the start point of the six homotopy paths. The end points of these paths are the six solutions of the target system.

Before caring out the computation, it is high time to implement four Mathematica functions to carry out linear homotopy computations in Mathematica environment.

5-9 Implementation of homotopy methods in Mathematica

5-9-1 Function for computing start system for polynomials

This function will compute the start system as well as the initial values for the computation of the homotopy paths in case of polynomial system.

**Input variables**

\( F \) - list of functions of the polynomial system to be solved, \( F = \{f_1(x), f_2(x), ..., f_n(x)\} \)

\( X \) - list of the independent variables, \( X = \{x_1, x_2, ..., x_n\} \)

\( d \) - list of the degrees of the polynomials, \( d = \{d_1, d_2, ..., d_n\} \)
\textbf{Output variables}

\begin{align*}
G & \quad \text{list of start system, } G = [g_1(x), g_2(x), \ldots, g_n(x)] \\
X0 & \quad \text{list of initial values, } X0 = \{[x_0, x_0, \ldots, x_0], [x_0, x_0, \ldots, x_0], \ldots, [x_0, x_0, \ldots, x_0]\} \\
& \quad \text{where } m \text{ is the number of the roots of } F
\end{align*}

\textbf{StartingSystem}[F_, X_, d_] := Module[{n, k, \phi, \Theta, Xi, X0},
    n = Length[X]
    \phi = Table[Random[Real, \{0, 2 \pi\}], \{n\}]
    \Theta = Table[Random[Real, \{0, 2 \pi\}], \{n\}]
    G = MapThread[\phi \mapsto \Theta \times \{x^2 \mapsto (x^2)^{n2} \times \{(x \times \\phi)^{\phi} + \{(x \times \\phi)^{\phi} \times \Theta\} \times \{(x \times \\phi)^{\phi} \times \Theta\}, (x, 0, \#2 - 1) \& , (\theta, d)\}]
    Xi = MapThread[Table[Exp[\phi \times \#1 + 2 \pi \times k / \#2], \{k, 0, \#2 - 1\} \& , (\theta, d)\}]
    X0 = Tuples[Xi];
    (G, X0)]

Now, let us create the start system with this function.
The list of the functions of the system we considered above is,

\begin{align*}
F & = \{f1[x, y], f2[x, y]\}
& = \{-1 + x^2 + y^2, -1 + x^2 + y^3\}
\end{align*}

The list of variables
\begin{align*}
X & = \{x, y\}
& = \{x, y\}
\end{align*}

The list of the degrees of the polynomials,
\begin{align*}
d & = \{d1, d2\}
& = \{2, 3\}
\end{align*}

Now, employing our function
\begin{align*}
sol & = \text{StartingSystem}[F, X, d];
\end{align*}

The start system is
\begin{align*}
G & = sol[[1]]
& = \{-0.419162 - 0.907912 \& , (0.657419 + 0.753525 \& ) \times x^2\},
& = \{0.609913 - 0.79316 \& , (0.354551 + 0.935037 \& ) \times y^3\}
\end{align*}

and the initial values are
\begin{align*}
X0 & = sol[[2]]
& = \{-0.910335 + 0.413873 \& , 0.799454 - 0.600727 \& \},
& = \{-0.910335 + 0.413873 \& , 0.120518 + 0.992711 \& \},
& = \{-0.910335 + 0.413873 \& , -0.919972 - 0.391984 \& \},
& = \{0.910335 - 0.413873 \& , 0.799454 - 0.600727 \& \},
& = \{0.910335 - 0.413873 \& , 0.120518 + 0.992711 \& \},
& = \{0.910335 - 0.413873 \& , -0.919972 - 0.391984 \& \}
\end{align*}

They are the solutions of the start system,
\begin{align*}
\text{Map}[G[[1]] / . \{x \rightarrow \#1[[1]], y \rightarrow \#1[[2]]\} \& , X0] // \text{Chop}
& = \{0, 0, 0, 0, 0\}
\end{align*}

\begin{align*}
\text{Map}[G[[2]] / . \{x \rightarrow \#1[[1]], y \rightarrow \#1[[2]]\} \& , X0] // \text{Chop}
& = \{0, 0, 0, 0, 0\}
\end{align*}
5-9-2 Function for direct path tracing

This function will compute the homotopy paths using Newton-Raphson method successively.

**Input variables**

- \( F \) - list of functions of the polynomial system to be solved, \( F = \{f_1(x), f_2(x), ..., f_n(x)\} \)
- \( G \) - list of the start system, \( G = \{g_1(x), g_2(x), ..., g_n(x)\} \)
- \( X \) - list of the independent variables \( X = \{x_1, x_2, ..., x_n\} \)
- \( X_0 \) - list of initial values, \( X_0 = \{x_0_1, x_0_2, ..., x_0_n\} \)
- \( m \) is the number of the roots of \( F \)
- \( \gamma \) - list of complex numbers, \( \{\gamma_1, \gamma_2, ..., \gamma_m\} \), it means \( \gamma_1 \) can be different for every \( g_i(x) \), if the start system is generated for polynomials all \( \gamma_i = 1 \).
- \( n \) - number of the subintervals in \([0, 1]\).

**Output variables**

- \( sol[1] \) - list of the \( i \)-th solutions corresponding to the \( i \)-th initial values, \( \{x_0_1, x_0_2, ..., x_0_n\} \), \( i = 1 ... m \)
- \( sol[2] \) - list of the path of \( i \)-th solutions in form of interpolating functions of the variables corresponding with the \( i \)-th initial values, \( \{x_0_1, x_0_2, ..., x_0_n\} \), \( i = 1 ... m \)

\[
\text{sol}[1] = \{\{\varphi_1, \varphi_2, ..., \varphi_n\} : \{\varphi_1, \varphi_2, ..., \varphi_n\}, \varphi_1 = \varphi_1(\lambda)\}
\]

\[
\text{LinearHomotopyFR}[F, G, X_0, X_0, \gamma_0, n, m] :=
\]

\[
\text{Module}\left[\{H, X0L, \lambda0, i, \beta, m, R, RR, j, k, sol\},
\text{Off}[\text{FindRoot}::\text{lstol}];
\lambda0 = \text{Table}[1/n, \{i, 0, n\}];
H = \text{Flatten}[\{1 - \beta\} \text{Thread}\[G \gamma + \beta F\]];
m = \text{Length}[X];
k = \text{Length}[X0];
\text{RR} = \{\};
\text{Do[}
\text{X0L} = \{X0[[j]]\};
\text{Do[\text{AppendTo}[X0L, \text{Map}[\text{H}[[2]] \&, \text{FindRoot}[H / . \beta \rightarrow \lambda0[[i + 1]], \text{MapThread}[\{\#1, \#2 \&, \{X, X0L[[i]]\}]]], \{i, 1, n\}];
R = \{\};
\text{Do[\text{AppendTo}[R, \text{Interpolation}[\text{MapThread}[\{\#1, \#2[[1]] \&, \{\lambda0, X0L\}]]], \{i, 1, m\};
\text{AppendTo}[RR, \text{Map}[\text{Chop}[\text{N}[\#1]] \&, R], R],
\{j, 1, k\}];
\text{sol} = \text{Transpose}[RR];
\{\text{sol}[1], \text{Table}[\text{MapThread}[\#1[\lambda] \rightarrow \#2[\lambda] \&, \{X, \text{sol}[[2, 1]]\}], \{i, 1, \text{Length}[X0][0]\}]\}
\]

**Remark:** It is reasonable to carry out computation with \( n = 100 \), however sometimes one may need more subintervals, but it also possible that less will enough to get satisfactory result.

In case of polynomial system, when we generate a start system in complex domain, we do not need to use complex \( \gamma \).

\[
\gamma = \{1, 1\};
\]

\[
\text{sol} = \text{LinearHomotopyFR}[F, G, X, X0, \gamma, 100];
\]
The first output gives the roots of the system,

\[
\text{Chop[sol[[1]], 10^{-8}]]}
\]

\[
\{(0, 1.), (-1 - 0.707107 \cdot, -1 + 0.707107 \cdot),
\{-1 + 0.707107 \cdot, -1 - 0.707107 \cdot}, \{0, 1.}, \{1, 0\}, \{1, 0\}\}
\]

The second output is the list of the interpolation functions of the paths,
sol[[2]]

\{x[\lambda] \rightarrow \text{InterpolatingFunction} \begin{array}{c} \text{Domain: \{0., 1.\}} \\
\text{Output: scalar} \end{array} \} [\lambda],

\{y[\lambda] \rightarrow \text{InterpolatingFunction} \begin{array}{c} \text{Domain: \{0., 1.\}} \\
\text{Output: scalar} \end{array} \} [\lambda],

5.9.3 Function for path tracing with integration
This function will compute the homotopy paths using numerical integration of the differential equation system employing Runge-Kutta method. In this implementation we compute the inverse of the Jacobian, namely the initial value problem will be expressed as

\[
\frac{dx(\lambda)}{d\lambda} = -H^{-1}_\lambda H
\]

with

\[ x(0) = x_0 \]

Here \( H^{-1}_\lambda \) is the inverse of the Jacobian \( H \) with respect to \( x_i, i = 1,..., n \), in case of \( n \) nonlinear equations with \( n \) variables. The inverse will be computed in symbolic way therefore the number of equations may be limited by \( n = 6 \times 8 \).

**Input variables**

- \( F \) - list of functions of the polynomial system to be solved, \( F = \{f_1(x), f_2(x),..., f_n(x)\} \)
- \( G \) - list of the start system, \( G = \{g_1(x), g_2(x),..., g_m(x)\} \)
- \( X \) - list of the independent variables \( X = \{x_1, x_2, ..., x_n\} \)
- \( X0 \) - list of initial values, \( X0 = \{[x_01, x_02, ..., x_01], [x_01, x_02, ..., x_02], ..., [x_01, x_02, ..., x_0n]\} \)
  
  where \( m \) is the number of the roots of \( F \)
- \( \gamma \) - list of complex numbers, \( \{\gamma_1, \gamma_2, ..., \gamma_n\} \), it means \( \gamma_i \) can be different for every \( g_i(x) \).
  
  If start system is generated for polynomials all \( \gamma_i = 1 \).
- \( p \) - indicating numerical precision of the computation., \( p = 0 \) standard precision, \( p = 1 \) high precision

**Output variables**

- \( \text{sol[1]} \) - list of the i-th solutions corresponding to the i-th initial values, \( \{x_01, x_02, ..., x_0n\} \), \( i = 1 \ldots m \)
- \( \text{sol[2]} \) - list of the path of i-th solutions in form of interpolating functions of the variables corresponding with the i-th initial values, \( \{x_01, x_02, ..., x_0n\} \), \( i = 1 \ldots m \),
  
  \( \{[\varphi_1, \varphi_2, ..., \varphi_n], [\varphi_1, \varphi_2, ..., \varphi_n], ..., [\varphi_1, \varphi_2, ..., \varphi_n]\} \), where \( \varphi_i = \varphi_i(\lambda) \)
LinearHomotopyNDS01[X_, F_, G_, X0_, γ_, P_] :=
Module[{H, n, Jλ, J, u, i, j, b, A, a, B, rhs, lhs, deqs, sol, pa, pp, wp},

Off[NDSolve::"precw"];
H = Flatten[(1 - λ) Thread[G γ] + λ F];
n = Length[X];
Jλ = Outer[D, H, Join[X, {λ}]]; J = Take[Jλ, {1, n}, {1, n}];
u = -Take[Jλ, {1, n}, {n + 1}];
A = Table[a[i, j], {i, 1, n}, {j, 1, n}];
B = Table[b[i, j], {i, 1, n}];

rhs = (Simplify[LinearSolve[A, B]] /.
(Join[Table[b[i, j, k] -> u[[j]], {i, 1, n}], Flatten[
Table[a[i, j] -> J[[i, j]], {i, 1, n}, {j, 1, n}]]]) /.
Map[# -> λ &] & /@ (X)); // Flatten;

lhs = Map[D[#, λ] &] & /@ (X);
deqs = MapThread[#1 == #2 &, {rhs, lhs}];

{pa, pp, wp} = If[p = 1,
{AccuracyGoal -> 20, PrecisionGoal -> 20, WorkingPrecision -> 30}, {(), {}}];
sol = Map[Flatten[NDSolve[Join[deqs, MapThread[#1[0] == #2 &, {X, H}]]],
Map[# &] &] & /@ (X); 
{Map[Chop[N[(Map[# &] & /@ X) /. # /. λ -> 1]] &] &@ sol, sol}]

Now we solve the same problem. First, we use standard precision.
sol = LinearHomotopyNDS01[X, F, G, X0, γ, 0];
sol[[1]]

{(-0.000290247 - 0.000300651 i, 1. - 4.4707 \times 10^{-6} i),
(-1. - 0.707107 i, 1. + 0.707107 i),
(-1. + 0.707107 i, 1. - 0.707107 i),
(-0.000125765 - 0.000316296 i, 1. + 1.95353 \times 10^{-6} i),
(1. - 6.5707 \times 10^{-8} i, 1. - 0.000194877 + 0.00026536 \times 10^{-14} i),
(1. - 6.23228 \times 10^{-9} i, 0.000258376 - 0.00031125 i)}

Chop[sol[[1]], 10^{-3}]

{(0, 1.), (-1. - 0.707107 i, -1. + 0.707107 i),
(-1. + 0.707107 i, 1. - 0.707107 i),
(0, 1.), {1, 0}, {1, 0}}

Using higher precision, the computation takes a considerably longer time,
sol = LinearHomotopyNDS01[X, F, G, X0, γ, 1];
sol1NDS = sol[[1]]

{(-2.20179 \times 10^{-9} + 6.65145 \times 10^{-9} i, 1.),
(-1. - 0.707107 i, -1. + 0.707107 i),
(-1. + 0.707107 i, -1. - 0.707107 i),
(1.90379 \times 10^{-8} - 1.44129 \times 10^{-8} i, 1.),
(1., -4.66527 \times 10^{-5} + 1.0209 \times 10^{-5} i),
(1., -3.03024 \times 10^{-8} + 2.28877 \times 10^{-8} i)}

but we have smaller values for the "tails",
Chop[sol1NDS, 10^{-7}]

{(0, 1.), (-1. - 0.707107 i, -1. + 0.707107 i),
(-1. + 0.707107 i, -1. - 0.707107 i),
(0, 1.), {1, 0}, {1, 0}}

The interpolation functions for the paths
sol2NDS = sol[[2]];
Remarks:

1) To get explicit form for the differential equation system, the linear equation system is solved in symbolic way. Therefore, the maximum number of equation can be about \( n = 6 - 8 \).

2) High precision solution (\( p = 1 \)) would take much longer time than standard precision does (\( p = 0 \)).

In order to avoid to compute the inverse in this implementation, we consider a new parameter, \( t \), namely

\[
dH(x(t), \lambda(t)) = \frac{\partial H}{\partial x} dx(t) + \frac{\partial H}{\partial \lambda} d\lambda(t) \equiv 0 \quad \lambda(t) \in [0, 1]
\]

this means with \( s(t) = \{x(t), \lambda(t)\} \) we get

\[
H_s(s(t)) \frac{ds}{dt} = 0
\]

where \( H_s \) is the Jacobian of the homotopy function respect to vector \( s \) having \( n + 1 \) dimensions.

Then the \( i^{th} \) derivative function can be expressed as

\[
\frac{ds_i}{dt} = (-1)^{i+1} \text{Det}(H_{s(1)}, ..., H_{s(n+1)}); \quad \text{H} = \text{Flatten}([1 - \lambda] \text{Thread}[G \ y] + \lambda F);
\]

Considering that

\[
\frac{dx_i}{dt} = \frac{dx}{dt} - \frac{dx}{d\lambda} \frac{d\lambda}{dt} = \frac{d\lambda}{dt} \frac{dx}{dt}
\]

for \( i = 1, 2, ..., n \).

\[
\frac{dx_i}{d\lambda} \frac{d\lambda}{dt} = \frac{dx}{dt}
\]

now the integration can be carried out with independent variable \( \lambda \) on \([0, 1]\). The new implementation considering these changes is the following,

**LinearHomotopyNDS02[X_, F_, G_, X0_, Y_, p_] :=**

```
Module[{H, n, Jx, rhs, rhs0, i, lhs, deqs, sol, pa, pp, wp},
   Off[NDSolve::"precw"];
   H = Flatten[(1 - \lambda) Thread[G y] + \lambda F];
   n = Length[X];
   Jx = Outer[D, H, Join[X, \{\lambda\}]];  
   rhs0 = Table[(-1)^i \text{Det}[\text{Transpose}[\text{Drop}[\text{Transpose}[Jx], \{i, i\}]], \{i, 1, n + 1\}];
   rhs = Flatten[\text{Drop}[\text{Map}[\# \& / \text{Last}[\text{rhs0}] \&, \text{rhs0}], \{n + 1, n + 1\}]] /. \text{Map}[\# \& \text{\rightarrow} \# \& \text{\&} \text{X}] // \text{Flatten};
   lhs = Flatten[\text{Drop}[\text{\&}, \{\lambda, 1\}] \&, \text{X}];
   deqs = \text{MapThread}[\#1 = \#2 \&, \{\text{lhs}, \text{rhs}\}];
   \{\text{pa}, \text{pp}, \text{wp}\} = \text{If}[p = 1, 
   \{\text{\{\\&\} \& \text{\rightarrow} 20, \text{\PrecisionGoal\} \& \text{\rightarrow} 20, \text{\WorkingPrecision\} \& \text{\rightarrow} 30}, \{\{}\, \{\}, \{\}\}\}];
   sol = \text{Map}[\text{\text{Flatten}[\text{NDSolve}][\text{Join}[\text{deqs}, \text{MapThread}[\#1[0] = \#2 \&, \{\text{X}, \text{H}\}]], 
   \text{Map}[\# \& \text{\&}, \text{X}], \{\lambda, 0, 1\}, \text{pa}, \text{pp}, \text{wp}] \&, \text{X0}];
   \{\text{Map}[\text{Chop}[\text{N}[\text{\{Map[\# \& \& X] / \\# \& \\cdot \lambda \\rightarrow 1\}]] \&, \text{sol}, \text{sol}]]
```
The input and output are the same as in case of the function `LinearHomotopyNDS01`.

```math
sol = LinearHomotopyNDS02[X, F, G, X0, γ, 1];
sol[[1]]
```

\[
\{-1.40248 \times 10^{-8}, 1.\}, \{-1. - 0.707107 \hat{i}, -1. + 0.707107 \hat{i}\},
\{-1. + 0.707107 \hat{i}, -1. - 0.707107 \hat{i}\}, \{2.30752 \times 10^{-8} - 1.24326 \times 10^{-5} \hat{i}, 1.\},
\{1., 1.15875 \times 10^{-8} - 9.99208 \times 10^{-9} \hat{i}\}, \{1., -3.1994 \times 10^{-8} + 2.33865 \times 10^{-8} \hat{i}\}\]

```math
Chop[sol[[1]], 10^{-7}]
```

\[
\{(0, 1.), \{-1. - 0.707107 \hat{i}, -1. + 0.707107 \hat{i}\},
\{-1. + 0.707107 \hat{i}, -1. - 0.707107 \hat{i}\}, \{0, 1.\}, \{1., 0\}, \{1., 0\}\}
\]

### 5.9.4 Function for visualization of paths

This function plots the homotopy paths computed by either direct method or by integration.

**Input variables**

- `X` - the list of the independent variables \(X = \{x_1, x_2, \ldots, x_n\}\)
- `sol` - list of the interpolation functions of the variables \(\{\{\varphi_1, \varphi_2, \ldots, \varphi_m\}, \{\varphi_1, \varphi_2, \ldots, \varphi_m\}\}\)
- `X0` - list of initial values \(X0 = \{[x_01, x_02, \ldots, x_0n], [x_01, x_02, \ldots, x_0n], \ldots, [x_01, x_02, \ldots, x_0n]\}\)

Where \(m\) is the number of the roots of \(F\).

```math
Paths[X, sol, X0] :=
GraphicsGrid[Table[ParametricPlot[{Re[X[[1]]][\lambda], Im[X[[1]]][\lambda]} /. sol[[j]],
\{\lambda, 0, 1\}, PlotStyle -> Thickness[Tiny], PlotRange -> All,
BaseStyle -> {FontSize -> 10, FontFamily -> "Times"}, Axes -> None,
FrameLabel -> ("Re", "Im"), Frame -> True, AspectRatio -> 0.6,
PlotLabel -> StringJoin[ToString[X[[1]]], " (\lambda )"], Epilog -> {PointSize[0.02],
Blue, Point[{Re[X0[[[j, 1]]], Im[X0[[[j, 1]]]]}], PointSize[0.02], Red, Point[
{Re[X[[1]]][\lambda] /. sol[[j]] /. \lambda -> 1, (Im[X[[1]]][\lambda] /. sol[[j]] /. \lambda -> 1)]}]
\{j, 1, Length[X0]\}, {i, 1, Length[X]]}]];
```

Let us plot the result of the solution of system above.
pNDS = Paths[X, sol[[2]], X0]

5- 10 Nonlinear homotopy

5- 10- 1 Quadratic Bezier homotopy function
The idea of this nonlinear homotopy function comes from the construction of the Bezier splines. Bezier curves are used to draw smooth curves along points on a path. In case of two points \((P_0, P_1)\) the point \(Q_0\) is running from \(P_0\) to \(P_1\) while the parameter \(\lambda\) is changing from 0 to 1, see Fig. 5.19

\[
\vec{Q}_0 = (1 - \lambda) \vec{P}_0 + \lambda \vec{P}_1, \quad \lambda \in [0, 1]
\]

![Fig. 5.19 Linear Bezier spline](Linear_and_Nonlinear_Homotopy_05.nb)

In case of three points \((P_0, P_1, P_2)\) the point \(Q_0\) is running from \(P_0\) to \(P_1\), while point \(Q_1\) is running from \(P_1\) to \(P_2\), see Fig. 5.20

\[
\vec{Q}_1 = (1 - \lambda) \vec{P}_1 + \lambda \vec{P}_2, \quad \lambda \in [0, 1]
\]

![Fig. 5.20 Quadratic Bezier spline](Linear_and_Nonlinear_Homotopy_05.nb)

and point \(R_0\) is running along a smooth path from \(P_0\) to \(P_2\) in a way,

\[
\vec{R}_0 = (1 - \lambda) \vec{Q}_0 + \lambda \vec{Q}_1, \quad \lambda \in [0, 1]
\]

or

\[
\vec{R}_0 = (1 - \lambda)^2 \vec{P}_0 + 2 (1 - \lambda) \lambda \vec{P}_1 + \lambda^2 \vec{P}_2, \quad \lambda \in [0, 1]
\]

Considering analogy between this Bezier curve construction and the quadratic homotopy function \textit{Nor et al} (2013) suggested the following casting,

\[
P_0 \sim G(x)
\]

\[
P_1 \sim H_1 (x, \lambda)
\]

\[
P_2 \sim F(x)
\]
and

\[ Q_0 \sim A(x, \lambda) \]
\[ Q_1 \sim B(x, \lambda) \]
\[ R_0 \sim H_2 (x, \lambda) \]

where \( H_1 (x, \lambda) \) is the linear homotopy function,

\[ H_1(x, \lambda) = (1 - \lambda) G(x) + \lambda F(x) \]

and \( H_2 (x, \lambda) \) is the quadratic Bezier homotopy function. Applying the analogy

\[ A(x, \lambda) = (1-\lambda) G(x) + \lambda H_1(x, \lambda) \]
\[ B(x, \lambda) = (1-\lambda) H_1(x, \lambda) + \lambda F(x) \]

then

\[ H_2 (x, \lambda) = (1-\lambda) A(x, \lambda) + \lambda B(x, \lambda) \]

or

\[ H_2 (x, \lambda) = (1-\lambda)^2 G(x) + 2 \lambda (1-\lambda) H_1 (x, \lambda) + \lambda^2 F(x) \]

\[ = (1 - \lambda)^2 G(x) + 2 \lambda (1-\lambda) ((1 - \lambda) G(x) + \lambda F(x)) + \lambda^2 F(x) \]

\[ \text{In[18]:=} pts1 = \{(0, -1), (2, 1), (4, 2), (6, 2)\}; \]
\[ pts2 = \{(2, -1), (3, 1), (4, -1), (6, 0)\}; \]
\[ \text{In[20]:=} \text{Animate[Graphics[[Blue, BezierCurve[pts1], Blue, BezierCurve[pts2], Thick, Red,} \]
\[ \text{BezierCurve[(1-t)^2 pts1 + 2 t (1-t) ((1-t) pts1 + t pts2) + t^2 pts2]], \{t, 0, 1\},} \]
\[ \text{AnimationRunning} \rightarrow \text{False, AnimationDirection} \rightarrow \text{ForwardBackward, SaveDefinitions} \rightarrow \text{True} \]

Fig. 5.21 Quadratic homotopy between two Bezier splines

The coefficients of \( H_2 (x, \lambda) \) can be computed also as the coefficients of Bezier function. Let us employ the variables

\[ H_0 = G(x) \text{ for } \lambda = 0 \]
\[ H_1 = H_1 (x, \lambda) \text{ for } \lambda \in (0, 1) \]
\[ H_2 = F(x) \text{ for } \lambda = 1 \]
then

\[ H_2(x, \lambda) = H_0 B_0^2(\lambda) + H_1 B_1^2(\lambda) + H_2 B_2^2(\lambda) \]

\[ = \sum_{i=0}^{n} H_i B_i^2(\lambda) \]

where \( B_i^n(\lambda) \) represents the \( i \)th Bernstein basis function of degree \( n \) at \( \lambda \),

\[ B_i^n(\lambda) = \binom{n}{i} (1 - \lambda)^{n-i} \lambda^i \]

where \( i = 0, 1, 2, \ldots, n \). In Mathematica has a built-in function for these basis functions, however it is only a numerical one, \texttt{BernsteinBasis[n, i, \lambda]}.

For linear homotopy

\[ \sum_{i=0}^{1} B_i^1(\lambda) = (1 - \lambda) + \lambda = 1 \]

\texttt{Plot[Sum[BernsteinBasis[1, i, 1], \{\lambda, 0, 1\}]}]

![Fig. 22 Numerical verification of the sum of the first degree of Bernstein basis](image)

For quadratic homotopy

\[ \sum_{i=0}^{2} B_i^2(\lambda) = (1 - \lambda)^2 + 2\lambda (1 - \lambda) + \lambda^2 = 1 \]

\texttt{Plot[Sum[BernsteinBasis[2, i, 1], \{\lambda, 0, 1\}]}]

![Fig. 23 Numerical verification of the sum of the second degree of Bernstein basis](image)

It goes without saying that

\[ H_2(x, 0) = G(x) \]

\[ H_2(x, 1) = F(x) \]

\textit{Remark}
A recursive construction of the quadratic homotopy function can be given by the De Casteljau’s algorithm. In our case see Fig. 24

\[ H_2(x, \lambda) \]

\[ H_1(x, \lambda) \]

\[ G(x) \]

\[ F(x) \]

Fig. 24 The De Casteljau’s algorithm

5-10-2 Implementation in Mathematica

The implementation of this quadratic homotopy function is easy, only one line should be changed.

NonLinearHomotopyFR[f_, g_, x_, x0_, y_, n_] :=
Module[{H, X0L, \lambda, i, \beta, m, R, RR, j, k, sol},
  Off[FindRoot::"lstol"]; \lambda 0 = Table[{1/(n), (i, 0, n)};
  H = Flatten[(1 - \beta)^2 Thread[g \gamma] + 2 \beta (1 - \beta) ((1 - \beta) Thread[g \gamma] + \beta F) + \beta^2 F; m = Length[X]; k = Length[X0]; RR = {};
  Do[
    X0L = (X0[[j]]);
    Do[AppendTo[X0L, Map[\#[[2]] \&,
      FindRoot[H / . \beta \rightarrow \lambda 0[[i + 1]], MapThread[\#1, \#2 \&, (X, X0L[[i]])]], {i, 1, n}];
      R = {};
      Do[AppendTo[R, Interpolation[MapThread[\#1, \#2 \&, (\lambda 0, X0L)]]], {i, 1, m}];
      AppendTo[RR, {Map[Chop[N[\#1]] \&, R], R}],
      {j, 1, k}];
    sol = Transpose[RR];
    {sol[[1]], Table[MapThread[\#1[\lambda] \rightarrow \#2[\lambda] \&, (X, sol[[2, i]])], {i, 1, Length[X0]})]}
  5-10-3 Comparing linear and quadratic homotopy

Let us consider this simple polynomial system

Clear[f1, f2]
f1[x_, y_] := x^2 - 2 x - y + 1/2
f2[x_, y_] := x^2 + 4 y^2 - 4
First we solve it with numerical Gröbner basis

```math
AbsoluteTiming[sol = NSolve[{f1[x, y], f2[x, y]}, {x, y}];
{0.0195764, Null}
```

The solution

```
sol
{(x \rightarrow 1.16077 - 0.654492 \imath, y \rightarrow -0.902513 - 0.210444 \imath),
 (x \rightarrow 1.16077 + 0.654492 \imath, y \rightarrow -0.902513 + 0.210444 \imath),
 (x \rightarrow -0.222215, y \rightarrow 0.993808), (x \rightarrow 1.90068, y \rightarrow 0.311219)}
```

Let us substitute back the solutions into the system and compute the mean of the error norms

```
Mean[Map[Norm[#]] & / sol]
4.71401 \times 10^{-15}
```

Now let us carry out the computation with linear homotopy. The system is

```
F = {f1[x, y], f2[x, y]}
\{\frac{1}{2} - 2 \times x^2 - y, -4 + x^2 + 4 \times y^2\}
```

The variables

```
X = {x, y}
{x, y}
```

The degree of the polynomials

```
d = {2, 2}
{2, 2}
```

We generate a start system

```
sol = StartingSystem[F, X, d];
G = sol[[1]]
{(-0.419162 - 0.907912 \imath) ((-0.657419 + 0.753525 \imath) + x^2),
 (0.609013 - 0.79316 \imath) ((-0.278254 + 0.960507 \imath) + y^2)}
```

with its solutions

```
X0 = sol[[2]]
{(-0.910335 + 0.413873 \imath, 0.799454 - 0.600727 \imath),
 (-0.910335 + 0.413873 \imath, -0.799454 + 0.600727 \imath),
 (0.910335 - 0.413873 \imath, 0.799454 - 0.600727 \imath),
 (0.910335 - 0.413873 \imath, -0.799454 + 0.600727 \imath)}
```

Now, we do not need \(\gamma\) trick,

```
y = {1, 1};
AbsoluteTiming[sol10L = LinearHomotopyFR[F, G, X, X0, y, 5];]
{0.0210878, Null}
```

The solution

```
sol10L[[1]]
{(-0.222215, 0.993808), (1.16077 + 0.654492 \imath, -0.902513 + 0.210444 \imath),
 (-0.222215, 0.993808), (1.16077 - 0.654492 \imath, -0.902513 - 0.210444 \imath)}
```
Back substitution

\[
\text{error10L} = \text{Map}[\text{F} /. \{x \to \text{#}[1], y \to \text{#}[2]\} \&, \text{sol10L}[1]]
\]

\[
\{0., 4.44089 \times 10^{-16}, 2.22045 \times 10^{-16} - 4.44089 \times 10^{-16} \text{\[\text{I}\]}, 0. - 4.44089 \times 10^{-16} \text{\[\text{I}\]}},
\{0., 4.44089 \times 10^{-16}, 0. + 0. \text{\[\text{I}\]}, 0. + 2.22045 \times 10^{-16} \text{\[\text{I}\]}},
\}
\]

The average of error norms

\[
\text{Mean}[\text{Map}[\text{Norm}[\text{#}] \&, \text{error10L}]]
\]

\[
4.44089 \times 10^{-16}
\]

Now we try quadratic homotopy

\[
\text{AbsoluteTiming}[\text{sol10NL} = \text{NonLinearHomotopyFR}[\text{F}, \text{G}, \text{X0}, \gamma, 5];]
\]

\[
\{0.0304908, \text{Null}\}
\]

The solution

\[
\text{sol10NL}[1]
\]

\[
\{-0.222215, 0.993808\}, \{1.16077 + 0.654492 \text{\[\text{I}\]}, -0.902513 + 0.210444 \text{\[\text{I}\]}},
\{-0.222215, 0.993808\}, \{1.16077 - 0.654492 \text{\[\text{I}\]}, -0.902513 - 0.210444 \text{\[\text{I}\]}},
\}
\]

Back substitution

\[
\text{error10NL} = \text{Map}[\text{F} /. \{x \to \text{#}[1], y \to \text{#}[2]\} \&, \text{sol10NL}[1]]
\]

\[
\{0., -4.44089 \times 10^{-16}, 0. + 0. \text{\[\text{I}\]}, -4.44089 \times 10^{-16} + 0. \text{\[\text{I}\]}},
\{1.11022 \times 10^{-16}, -4.44089 \times 10^{-16}, 0. + 0. \text{\[\text{I}\]}, 0. + 2.22045 \times 10^{-16} \text{\[\text{I}\]}},
\}
\]

The average of error norms

\[
\text{Mean}[\text{Map}[\text{Norm}[\text{#}] \&, \text{error10NL}]]
\]

\[
3.91995 \times 10^{-16}
\]

In order to improve the result of the linear homotopy we need more steps,

\[
\text{AbsoluteTiming}[\text{sol10L} = \text{LinearHomotopyFR}[\text{F}, \text{G}, \text{X0}, \gamma, 300];]
\]

\[
\{0.53911, \text{Null}\}
\]

The solution

\[
\text{sol10L}[1]
\]

\[
\{1.90068, 0.311219\}, \{1.16077 + 0.654492 \text{\[\text{I}\]}, -0.902513 + 0.210444 \text{\[\text{I}\]}},
\{-0.222215, 0.993808\}, \{1.16077 - 0.654492 \text{\[\text{I}\]}, -0.902513 - 0.210444 \text{\[\text{I}\]}},
\}
\]

Back substitution

\[
\text{error10L} = \text{Map}[\text{F} /. \{x \to \text{#}[1], y \to \text{#}[2]\} \&, \text{sol10L}[1]]
\]

\[
\{-4.44089 \times 10^{-16}, -7.21645 \times 10^{-16},
\{0. + 0. \text{\[\text{I}\]}, 0. + 0. \text{\[\text{I}\]}}, \{0., 4.44089 \times 10^{-16}, 0. + 0. \text{\[\text{I}\]}, 0. + 0. \text{\[\text{I}\]}},
\}
\]

Now The average of error norms is smaller but the running time longer

\[
\text{Mean}[\text{Map}[\text{Norm}[\text{#}] \&, \text{error10L}]]
\]

\[
3.22858 \times 10^{-16}
\]

In this example the quadratic homotopy gives higher precision than the linear one or linear homotopy requires longer computation time to reach the same error limit. However, until now, only restricted computational experiments are at our disposal!
5-11 Examples

5-11-1 Non-polynomial system

As illustration, first let us see the solution of a non-polynomial system. Let us consider the following equations,

\[
\begin{align*}
\text{f}_1(x, y) &= x - \sin(2x + 3y) + \cos(3x - 5y) \\
\text{f}_2(x, y) &= y - \sin(x - 2y) + \cos(x + 3y)
\end{align*}
\]

We generate the contour plot of the equations in the region of \([-2, 2]^2\).

\[
\text{f}_1[x_\text{, } y_] := x - \sin[2x + 3y] + \cos[3x - 5y] \\
\text{f}_2[x_\text{, } y_] := y - \sin[x - 2y] + \cos[x + 3y]
\]

\[
\text{ContourPlot}[\{\text{f}_1[x, y] == 0, \text{f}_2[x, y] == 0\}, \{x, -2, 2.5\}, \{y, -2, 2\}]
\]

The target system

\[
\mathbf{F} = \{\text{f}_1[x, y], \text{f}_2[x, y]\}
\]

\[
\{x + \cos[3x - 5y] - \sin[2x + 3y], y + \cos[x + 3y] - \sin[x - 2y]\}
\]

Let us try to find a root starting from \(x = 1.5, y = 1.5\) with Newton-Raphson method

\[
\text{FindRoot}[\mathbf{F}, \{(x, 2), (y, -2)\}]
\]

\[
\{x \to 1.06913, y \to -0.286591\}
\]

It is not a solution

\[
\mathbf{F} /. \%
\]

\[
\{0.0395526, -0.30587\}
\]

Let us employ fixpont homotopy, the start system

\[
\mathbf{G} = \{x - 2, y + 2\}
\]

\[
\{-2 + x, 2 + y\}
\]

Then

\[
\mathbf{X} = \{x, y\};
\]

\[
\mathbf{X}0 = \{(2, -2)\};
\]

\[
\gamma = \{1, 1i\};
\]

\[
\text{sol} = \text{LinearHomotopyFR}[\mathbf{F}, \mathbf{G}, \mathbf{X}0, \gamma, 400];
\]
It is a solution

\[
F / . \text{MapThread}[^1 -> ^2 & , \{X, \text{Flatten[sol[[1]]]]\}] // \text{Chop}
\]

\{0, 0\}

\text{Paths}[X, \text{sol[[2]]}, X0]

\begin{align*}
\text{Figure 5.26.} & \quad \text{Homotopy paths are straight lines, values are real in case of } x_0 = 1.5 \text{ and } y_0 = 1.5 \\
5-11-2 \quad 3D \text{ Resection Problem} & \\
The three-dimensional resection problem concerns itself with the determination of position and orientation of point \(P_0\) connected by angular observations, \(\varphi_{ij}, i, j = 1, 2, 3\) to three known stations, \(P_i, i = 1, 2, 3\), see Fig. 5.23. It means, that the space angles, \(\varphi_{ij}\) as well as the distances \(S_{ij}, i, j = 1, 2, 3\) are known and the distances \(S_i, i = 1, 2, 3\) should be computed.

\begin{align*}
S_{1,2}^2 & = S_1^2 + S_2^2 - 2 S_1 S_2 \cos (\varphi_{1,2}) \\
S_{2,3}^2 & = S_2^2 + S_3^2 - 2 S_2 S_3 \cos (\varphi_{2,3}) \\
S_{3,1}^2 & = S_3^2 + S_1^2 - 2 S_3 S_1 \cos (\varphi_{3,1})
\end{align*}

\text{Grunert} proposed the following distance equations,

This is a system of polynomial equations for the variables \(S_1, S_2\) and \(S_3\) representing distances.

Let us introduce the following variables

\[
\cos (\varphi_{ij}) = f_{ij}, \quad S_i = x_i \quad \text{and} \quad S_{ij} = \sqrt{d_{ij}}
\]

then our system can be written in the following form,
\[p_1 = x_1^2 - 2f_1x_1x_2 + x_2^2 - d_{12}; \]
\[p_2 = x_2^2 - 2f_2x_2x_3 + x_3^2 - d_{23}; \]
\[p_3 = x_3^2 - 2f_3x_1x_3 + x_1^2 - d_{31}; \]

Data are
\[
data = \text{SetPrecision}[\{d_{12} \rightarrow 1560.3302^2, d_{23} \rightarrow 755.8681^2, d_{31} \rightarrow 1718.1090^2, f_{12} \rightarrow \cos[1.843620], f_{23} \rightarrow \cos[1.768989], f_{31} \rightarrow \cos[2.664537]\}, 20];
\]
\[X = \{x_1, x_2, x_3\}; \]
\[F = \{p_1, p_2, p_3\} \text{/} . \text{data} \]
\[
\{-2.4346303330320403911 \times 10^8 + x_1^2 + 0.53890348757379025191x_1x_2 + x_2^2, \\
-571.336.58459761005361 + x_2^2 + 0.39379541340558754658x_2x_3 + x_3^2, \\
-2.95189853580996396 \times 10^6 + x_1^2 + 1.7767014278358685964x_1x_3 + x_3^2\}
\]
\[d = \{2, 2, 2\}; \]

The start system
\[ss = \text{StartingSystem}[F, X, d]; \]
\[G = ss[[1]] \]
\[
\{(-0.419162 - 0.907912 \, \text{i}) \{(-0.278254 + 0.960507 \, \text{i}) + x_1^2\}, \\
(0.609013 - 0.79316 \, \text{i}) \{(-0.187942 + 0.98218 \, \text{i}) + x_2^2\}, \\
(-0.910335 + 0.413873 \, \text{i}) \{(-0.983559 + 0.18059 \, \text{i}) + x_3^2\}\}
\]
The initial values
\[X_0 = ss[[2]] \]
\[
\{0.799454 - 0.600727 \, \text{i}, 0.770695 - 0.637204 \, \text{i}, -0.995881 + 0.0906683 \, \text{i}\}, \\
\{0.799454 - 0.600727 \, \text{i}, 0.770695 - 0.637204 \, \text{i}, 0.995881 - 0.0906683 \, \text{i}\}, \\
\{0.799454 - 0.600727 \, \text{i}, -0.770695 + 0.637204 \, \text{i}, -0.995881 + 0.0906683 \, \text{i}\}, \\
\{0.799454 - 0.600727 \, \text{i}, -0.770695 + 0.637204 \, \text{i}, 0.995881 - 0.0906683 \, \text{i}\}, \\
\{-0.799454 + 0.600727 \, \text{i}, 0.770695 - 0.637204 \, \text{i}, -0.995881 + 0.0906683 \, \text{i}\}, \\
\{-0.799454 + 0.600727 \, \text{i}, 0.770695 - 0.637204 \, \text{i}, 0.995881 - 0.0906683 \, \text{i}\}, \\
\{-0.799454 + 0.600727 \, \text{i}, -0.770695 + 0.637204 \, \text{i}, -0.995881 + 0.0906683 \, \text{i}\}, \\
\{-0.799454 + 0.600727 \, \text{i}, -0.770695 + 0.637204 \, \text{i}, 0.995881 - 0.0906683 \, \text{i}\}\}
\]
The upper bound of the number of the estimated isolated roots
\[\text{Length}[X_0]\]
8

Let
\[\gamma = \{1, 1, 1\}; \]

Employing direct path tracing algorithm
\[\text{AbsoluteTiming}[\text{sol} = \text{LinearHomotopyFR}[F, G, X_0, \gamma, 100]];\]
\[\{0.523701, \text{Null}\}\]
The solutions are
sol[[1]]
{(-1580.11, 770.958, -153.711),
 (-22456.5 + 1735.3 i, 4375.48 + 22037.5 i, 20757.3 - 8626.43 i),
 (-1324.24, -542.261, -430.529),
 (22456.5 + 1735.3 i, -4375.48 + 22037.5 i, -20757.3 - 8626.43 i),
 (-22456.5 + 1735.3 i, 4375.48 - 22037.5 i, 20757.3 + 8626.43 i),
 (1324.24, 542.261, 430.529),
 (-22456.5 + 1735.3 i, -4375.48 - 22037.5 i, -20757.3 + 8626.43 i),
 (-1580.11, -770.958, 153.711)}

Selecting real, positive solution

solR = Select[sol[[1]], (Im[#1] == 0 && #1 > 0) &&
 (Im[#2] == 0 && #2 > 0) && (Im[#3] == 0 && #3 > 0) &] // Flatten

{1324.24, 542.261, 430.529}

errorR = F /. (x1 -> solR[[1]], x2 -> solR[[2]], x3 -> solR[[3]])

{3.49246 \times 10^{-10}, 0., 4.65661 \times 10^{-10}}

Norm[errorR]

5.82077 \times 10^{-10}

The position of this solution

p = Position[sol[[1]], solR] // Flatten // First

6

Homotopy solution paths of the 3D resection problem:

In order to print the trajectories in one column we use a modified function

PathsV[X_, sol_, X0_] := GraphicsGrid[
 Transpose[Table[Flatten[ParametricPlot[{Re[X[[i]]]}, Im[X[[i]]]], /. sol[[j]],
 (\lambda, 0, 1), PlotStyle -> Thickness[Tiny], PlotRange -> All,
 BaseStyle -> {FontSize -> 10, FontFamily -> "Times"}, Axes -> False,
 FrameLabel -> ("Re", "Im"), Frame -> True, AspectRatio -> 0.6,
 PlotLabel -> StringJoin[ToString[X[[i]]], "(\lambda) "],
 Epilog -> {PointSize[0.02], Blue, Point[{Re[X0[[j, i]]]], Im[X0[[j, i]]]}],
 PointSize[0.02], Red, Point[{(Re[X[[i]]])/. sol[[j]]} /. \lambda -> 1,
 (Im[X[[i]]])/. sol[[j]] /. \lambda -> 1])]]},
 {j, 1, Length[X0]}, {i, 1, Length[X0]}], ImageSize -> Small];
PathsV[X, {sol[[2]][[1]]}, {X0[[1]]}]

AbsoluteTiming[sol = NonLinearHomotopyFR[F, G, X, X0, γ, 100];]
{0.726011, Null}

The solutions are
sol[[1]]

{(-1580.11, 770.958, -153.711),
 {(-22456.5 - 1735.3 \ i, 4375.48 + 22037.5 \ i, 20757.3 - 8626.43 \ i),
 {(-1324.24, -542.261, -430.529),
 {22456.5 - 1735.3 \ i, -4375.48 + 22037.5 \ i, -20757.3 - 8626.43 \ i),
 {(-22456.5 + 1735.3 \ i, 4375.48 - 22037.5 \ i, 20757.3 + 8626.43 \ i),
 {1324.24, 542.261, 430.529},
 {22456.5 + 1735.3 \ i,
 -4375.48 - 22037.5 \ i, -20757.3 + 8626.43 \ i),
 {1580.11, -770.958, 153.711}}}

Selecting real, positive solution
solR = Select[sol[[1]], (Im[H[[1]]] <= 0) && (Im[H[[2]]] <= 0) && (Im[H[[3]]] <= 0) && Flatten[
{1324.24, 542.261, 430.529}]

Fig. 5.28 The real root of the polynomial
errorR = F /. \{x1 \to solR[[1]], x2 \to solR[[2]], x3 \to solR[[3]]\}
\{-2.32831 \times 10^{-10}, 0., -2.32831 \times 10^{-10}\}

Norm[errorR]
3.29272 \times 10^{-10}

Now, there is the nonlinear homotopy somewhat better, but the price for it a longer running time. Let us see how can parallel computing reduce the running time.

Having access to more processors and/or more cores, homotopy solution can be computed in parallel way, since every homotopy path belonging to different start values can be traced independently, simultaneously. Mathematica can utilize such hardware configuration. You need a very small modification in calling our function.

Let us compute the GPS solution with single and with double core. The computation time can be reduced considerably, especially when the original running time is high.

Here, in case of direct path tracing we used 1000 subinterval instead of 100 in order to stress the difference in the running time, between one and double core computations.

LaunchKernels[2 \$ProcessorCount] // Quiet;
DistributeDefinitions[NonLinearHomotopyFR, X, F, G, X0, \[Gamma]];

In order to compare non parallel and parallel computation we use here 1000 steps

AbsoluteTiming[sol = NonLinearHomotopyFR[F, G, X0, \[Gamma], 1000];]
\{7.08203, Null\}

AbsoluteTiming[sol = ParallelMap[NonLinearHomotopyFR[F, G, X, \{\[Pi]\}, \[Gamma], 1000] & , X0];]
\{1.745, Null\}

sol1 = Map[Flatten[H[[1]]] & , sol]
\{-1580.11, 770.958, -153.711\},
\{-22456.5 - 1735.3 i, 4375.48 + 22037.5 i, 20757.3 - 8626.43 i\},
\{-1324.24, -542.261, -430.529\},
\{22456.5 - 1735.3 i, -4375.48 + 22037.5 i, -20757.3 - 8626.43 i\},
\{-22456.5 + 1735.3 i, 4375.48 - 22037.5 i, 20757.3 + 8626.43 i\},
\{1324.24, 542.261, 430.529\}, \{22456.5 + 1735.3 i, -4375.48 - 22037.5 i, -20757.3 + 8626.43 i\}, \{1580.11, -770.958, 153.711\}\}

solR = Select[sol1, (Im[H[[1]]] = 0 \[And] \[Pi][1] > 0) \[And] (Im[H[[2]]] = 0 \[And] \[Pi][2] > 0) \[And] (Im[H[[3]]] = 0 \[And] \[Pi][3] > 0) \&] // Flatten
\{1324.24, 542.261, 430.529\}

Now let us employ the path tracing algorithm by integration with high precision,

AbsoluteTiming[sol = LinearHomotopyNDS02[X, F, G, X0, \[Gamma], 1];]
\{18.8793, Null\}

DistributeDefinitions[LinearHomotopyNDS02, X, F, G, X0, \[Gamma]]; AbsoluteTiming[sol = ParallelMap[LinearHomotopyNDS02[X, F, G, \{\[Pi]\}, \[Gamma], 1] & , X0];]
\{8.64598, Null\}

We had 4 cores representing 2 threads each. So we have 8 threads, and one should realize that the total task could be distributed into 8 subtasks, so the total power (100%) of the processor could be utilized.

Length[X0]
5-11-3 GPS Positioning 4-point Problem

Throughout history, position determination has been one of the most important tasks of mountaineers, pilots, sailor, civil engineers etc. In modern times, Global Positioning System (GPS) employing Global Navigation Satellite Systems (GNSS) provide an ultimate method to accomplish this task. If one has a hand held GPS receiver, the receiver measures the travel time of the signal transmitted from the satellites. Then this distance can be computed by multiplying the measured time by the speed of light in vacuum. The distance of the receiver from the i-th satellite, \( d_i \) is related to the unknown position of the receiver, \( \{X, Y, Z\} \)

\[
d_i = \sqrt{(a_i - X)^2 + (b_i - Y)^2 + (c_i - Z)^2} + \xi
\]

where \( \{a_i, b_i, c_i\} \), \( i = 0 \ldots 3 \) are the coordinates of the \( i^{th} \) satellite.

The distance is influenced also by the satellite and receiver' clock biases. The satellite clock biases can be modeled while the receiver' clock biases have to be considered as an unknown variable, \( \xi \). This means, we have four unknowns, consequently we need four satellite signals as minimum observation. Let us employ \( x_1, x_2, x_3 \) and \( x_4 \) variables for the four unknowns, \( \{X, Y, Z, \xi\} \) then our equations are, \( e_i = 0, \ i = 1...4 \), where

\[\text{Clear}[d, b, f1, f2]\]

\[\begin{align*}
e1 &= (x1 - a0)^2 + (x2 - b0)^2 + (x3 - c0)^2 - (x4 - d0)^2; \\
e2 &= (x1 - a1)^2 + (x2 - b1)^2 + (x3 - c1)^2 - (x4 - d1)^2; \\
e3 &= (x1 - a2)^2 + (x2 - b2)^2 + (x3 - c2)^2 - (x4 - d2)^2; \\
e4 &= (x1 - a3)^2 + (x2 - b3)^2 + (x3 - c3)^2 - (x4 - d3)^2;
\end{align*}\]

or

\[\{f1, f2, f3, f4\} = \{e1, e2, e3, e4\} \text{ } \text{ } \text{ // Expand}\]

\[\begin{align*}
x1^2 + x2^2 + x3^2 - x4^2 - 2 x1 a0 + a^2 + 2 x2 b0 + b^2 - 2 x3 c0 + c^2 + 2 x4 d0 - d^2, \\
x1^2 + x2^2 + x3^2 - x4^2 - 2 x1 a1 + a^2 + 2 x2 b1 + b^2 - 2 x3 c1 + c^2 + 2 x4 d1 - d^2, \\
x1^2 + x2^2 + x3^2 - x4^2 - 2 x1 a2 + a^2 + 2 x2 b2 + b^2 - 2 x3 c2 + c^2 + 2 x4 d2 - d^2, \\
x1^2 + x2^2 + x3^2 - x4^2 - 2 x1 a3 + a^2 + 2 x2 b3 + b^2 - 2 x3 c3 + c^2 + 2 x4 d3 - d^2\}
\]

Now let us create the start system in a different way. We consider the univariable parts of the equations as start system. Namely,

\[
g1 = a0^2 + b0^2 + c0^2 - d0^2 + \text{Coefficient}[f1, x1, 1] x1 + \text{Coefficient}[f1, x1, 2] x1^2 \\
x1^2 - 2 x1 a0 + a0^2 + b0^2 + c0^2 - d0^2 \]

\[
g2 = a1^2 + b1^2 + c1^2 - d1^2 + \text{Coefficient}[f2, x2, 1] x2 + \text{Coefficient}[f2, x2, 2] x2^2 \\
x2^2 - a1^2 + 2 x2 b1 + b1^2 + c1^2 - d1^2 \]

\[
g3 = a2^2 + b2^2 + c2^2 - d2^2 + \text{Coefficient}[f3, x3, 1] x3 + \text{Coefficient}[f3, x3, 2] x3^2 \\
x3^2 + a2^2 + b2^2 - 2 x3 c2 + c2^2 - d2^2 \]

\[
g4 = a3^2 + b3^2 + c3^2 - d3^2 + \text{Coefficient}[f4, x4, 1] x4 + \text{Coefficient}[f4, x4, 2] x4^2 \\
x4^2 + a3^2 + b3^2 + c3^2 + 2 x4 d3 - d3^2 \]

The observation data are,
data = \{a_0 \rightarrow 1.483230866 \times 10^7, \\
a_1 \rightarrow -1.57998540510^7, \\
a_2 \rightarrow 1.98481891 \times 10^6, \\
a_3 \rightarrow -1.24802731910^7, \\
b_0 \rightarrow -2.04667158910^7, \\
b_1 \rightarrow -1.33011291710^7, \\
b_2 \rightarrow -1.18676729610^7, \\
b_3 \rightarrow -2.33825605310^7, \\
c_0 \rightarrow -7.4286347510^6, \\
c_1 \rightarrow 1.713383824 \times 10^7, \\
c_2 \rightarrow 2.371692013 \times 10^7, \\
c_3 \rightarrow 3.27847268 \times 10^6. \\
d_0 \rightarrow 2.431076406 \times 10^7, \\
d_1 \rightarrow 2.2914600784 \times 10^7, \\
d_2 \rightarrow 2.0628809405 \times 10^7, \\
d_3 \rightarrow 2.3422377972 \times 10^7\};

The target system
\[ F = \{f_1, f_2, f_3, f_4\} / . \text{data} \]
\[ \{1.03055 \times 10^{14} - 2.96646 \times 10^7 x_1 x_1^2 + 4.09334 \times 10^7 x_2 x_2^2 + \\
1.48573 \times 10^7 x_3 x_3^2 + 4.86215 \times 10^7 x_4 - x_4^2, \\
1.95045 \times 10^{14} + 3.15997 \times 10^7 x_1 + \\
x_1^2 + 2.66023 \times 10^7 x_2 + x_2^2 + 3.42677 \times 10^7 x_3 + x_3^2 + 4.58292 \times 10^7 x_4 - x_4^2, \\
2.81726 \times 10^{14} - 3.96964 \times 10^6 x_1 + x_1^2 + 2.37353 \times 10^7 x_2 + x_2^2 - 4.74338 \times 10^7 x_3 + \\
x_3^2 + 4.12576 \times 10^7 x_4 - x_4^2, \\
1.64642 \times 10^{14} + 2.49605 \times 10^7 x_1 + x_1^2 + \\
4.67651 \times 10^7 x_2 + x_2^2 - 6.55695 \times 10^6 x_3 + x_3^2 + 4.68448 \times 10^7 x_4 - x_4^2\} \]

The start system consisting of univariate polynomials,
\[ G = \{g_1, g_2, g_3, g_4\} / . \text{data} \]
\[ \{1.03055 \times 10^{14} - 2.96646 \times 10^7 x_1 x_1^2 + 1.95045 \times 10^{14} + 2.66023 \times 10^7 x_2 + x_2^2, \\
2.81726 \times 10^{14} - 4.74338 \times 10^7 x_3 + x_3^2, \\
1.64642 \times 10^{14} + 4.68448 \times 10^7 x_4 - x_4^2\} \]

Let us compute the initial values as the solution of the start system,
\[ X = \{x_1, x_2, x_3, x_4\}; \]
\[ X0 = \text{Transpose}[\text{Partition}[\text{Map}[[\#][[2]] \&, \text{Flatten}[\text{MapThread}[\text{NSolve}[\#1 == 0, \#2 \&, \{G, X\}]]], 2]] \]
\[ \{(4.01833 \times 10^6, -1.33011 \times 10^7 - 4.25733 \times 10^6 \text{ \_} 1, 6.96083 \times 10^6, -3.28436 \times 10^6), \\
(2.56463 \times 10^7, -1.33011 \times 10^7 + 4.25733 \times 10^6 \text{ \_} 1, 4.0473 \times 10^7, 5.01291 \times 10^7)\} \]

Employing the path tracing algorithm by integration,
\[ y = \{1, 1, 1, 1\}; \]
\[ \text{sol} = \text{LinearHomotopyFR}[F, G, X, X0, y, 500]; // \text{AbsoluteTiming} \]
\[ \{1.05761, \text{Null}\} \]
and gives practically one solution
\[ \text{sol}[[1]] \]
\[ \{1.11159 \times 10^6, -4.34826 \times 10^6, 4.52735 \times 10^6, 100.001\}, \\
\{1.11159 \times 10^6, -4.34826 \times 10^6, 4.52735 \times 10^6, 100.001\}\}\]
\[ \text{solR = sol}[[1]][[1]] \]
\[ \{1.11159 \times 10^6, -4.34826 \times 10^6, 4.52735 \times 10^6, 100.001\} \]
or
\[ \text{error} = F / . \{x_1 \rightarrow \text{solR}[[1]], x_2 \rightarrow \text{solR}[[2]], x_3 \rightarrow \text{solR}[[3]], x_4 \rightarrow \text{solR}[[4]]\} \]
\[ \{0.03125, 0., -0.03125, -0.00195313\}\]
\[ \text{Norm[error]} \]
\[ 0.0442373 \]
sol = NonLinearHomotopyFR[F, G, X, X0, γ, 500]; // AbsoluteTiming
{1.51785, Null}
and gives practically one solution

sol[[1]]

{{1.11159 \times 10^6, -4.34826 \times 10^6, 4.52735 \times 10^6, 100.001},
 {1.11159 \times 10^6, -4.34826 \times 10^6, 4.52735 \times 10^6, 100.001}}

solR = sol[[1]][[1]]

{1.11159 \times 10^6, -4.34826 \times 10^6, 4.52735 \times 10^6, 100.001}

error = F /. {x1 \to solR[[1]], x2 \to solR[[2]], x3 \to solR[[3]], x4 \to solR[[4]]}

{0., 0., -0.03125, 0.}
Norm[error]
0.03125

The built function NSolve employing numerical Groebner basis provides the same result,

AbsoluteTiming[sol = NSolve[{F, X}];]

{0.0336037, Null}
sol

{{x1 \to -2.89212 \times 10^6, x2 \to 7.56878 \times 10^6, x3 \to -7.20951 \times 10^6, x4 \to 5.74799 \times 10^7},
 {x1 \to 1.11159 \times 10^6, x2 \to -4.34826 \times 10^6, x3 \to 4.52735 \times 10^6, x4 \to 100.001}}

Norm[F /. %[[2]]]

0.0690258

The nonlinear homotopy provide the longest running time but the most precise solution.

Let us see the parallel computation

AbsoluteTiming[sol = LinearHomotopyFR[F, G, X, X0, γ, 1000];]

{2.04255, Null}
sol[[1]]

{{1.11159 \times 10^6, -4.34826 \times 10^6, 4.52735 \times 10^6, 100.001},
 {1.11159 \times 10^6, -4.34826 \times 10^6, 4.52735 \times 10^6, 100.001}}

DistributeDefinitions[LinearHomotopyFR, X, F, G, X0, γ];

Now, we employ a pure function for the elements of the list of the initial values (start values),

AbsoluteTiming[sol = ParallelMap[LinearHomotopyFR[F, G, X, {#, γ, 1000} &], X0];]

{1.02976, Null}
sol[[1]][[1]]

{{1.11159 \times 10^6, -4.34826 \times 10^6, 4.52735 \times 10^6, 100.001}}

Now employing the path tracing algorithms by integration with high precision, and using numerical inverse,

AbsoluteTiming[sol = LinearHomotopyNDS02[X, F, G, X0, γ, 1];]

{55.3815, Null}
DistributeDefinitions[LinearHomotopyNDS02, X, F, G, X0, γ];

AbsoluteTiming[sol = ParallelMap[LinearHomotopyNDS02[X, F, G, {#, γ, 1} &], X0];]

{48.6577, Null}
\[
\text{sol}[[1]][[1]]
\]  
\[
\{1.11159 \times 10^6 + 1.78522 \times 10^{-10}, -4.34826 \times 10^6 + 8.87828 \times 10^{-9}, \\
4.52735 \times 10^6 - 3.88685 \times 10^{-9}, 100.001 - 7.19922 \times 10^{-10}\}
\]

Now the reduction of the computation time is very modest because, we have only two jobs!

**5-11-4 GPS Positioning N-point Problem**

In case of \( m > 4 \) satellites, the two representations,

\[
f_i = (x_1-a)^2 + (x_2-b)^2 + (x_3-c)^2 - (x_4-d)^2
\]

and

\[
g_i = d_i - \sqrt{(x_1-a)^2 + (x_2-b)^2 + (x_3-c)^2} - x_4
\]

will be not equivalent in least square sense, namely

\[
\min_{x_1, x_2, x_3, x_4} \sum_{i=1}^{m} f_i^2 \neq \min_{x_1, x_2, x_3, x_4} \sum_{i=1}^{m} g_i^2
\]

Let us consider six satellites with the following numerical values,

\[
data = \{a_0 \rightarrow 14177553.47, a_1 \rightarrow 15097199.81, \\
a_2 \rightarrow 23460342.33, a_3 \rightarrow -8206488.95, a_4 \rightarrow 1399988.07, a_5 \rightarrow 6995655.48, \\
b_0 \rightarrow -18814768.09, b_1 \rightarrow -4636088.67, b_2 \rightarrow -9433518.58, \\
b_3 \rightarrow -18217989.14, b_4 \rightarrow -17563734.90, b_5 \rightarrow -23537808.26, \\
c_0 \rightarrow 12243866.38, c_1 \rightarrow 21326706.55, c_2 \rightarrow 8174941.25, \\
c_3 \rightarrow 17605231.99, c_4 \rightarrow 19705591.18, c_5 \rightarrow -9927906.48, \\
d_0 \rightarrow 21119278.32, d_1 \rightarrow 22527064.18, d_2 \rightarrow 23674159.88, \\
d_3 \rightarrow 20951647.38, d_4 \rightarrow 20155401.42, d_5 \rightarrow 24222110.91\};
\]

The number of the equations,

\( m = 6 \);

However, if we employ the norm of the distance instead of the residuum of the equations, namely

\[
en = d_i - \sqrt{(x_1-a)^2 + (x_2-b)^2 + (x_3-c)^2} - x_4;
\]

since

\[
\text{Map[D[en^2,\#]&,\{x_1,\,x_2,\,x_3,\,x_4\}]}\]

\[
\{-\text{\{2 (x_1-a)^2 - x_4 - \sqrt{(x_1-a)^2 + (x_2-b)^2 + (x_3-c)^2} + (2, 2, 2, 1)\}\} / \\
\{(x_1-a)^2 + (x_2-b)^2 + (x_3-c)^2\}\}, \\
-\text{\{2 (x_2-b)^2 - x_4 - \sqrt{(x_1-a)^2 + (x_2-b)^2 + (x_3-c)^2} + (2, 2, 2, 1)\}\} / \\
\{(x_1-a)^2 + (x_2-b)^2 + (x_3-c)^2\}\}, \\
-\text{\{2 (x_3-c)^2 - x_4 - \sqrt{(x_1-a)^2 + (x_2-b)^2 + (x_3-c)^2} + (2, 2, 2, 1)\}\} / \\
\{(x_1-a)^2 + (x_2-b)^2 + (x_3-c)^2\}\}, \\
-2 (x_1-a)^2 + (x_2-b)^2 + (x_3-c)^2 + (2, 2, 2, 1)\} / \\
\{(x_1-a)^2 + (x_2-b)^2 + (x_3-c)^2\}\}
\]

then in general form,
\( v = \text{Map}\left[ \sum_{i=0}^{m} \# \, \% \right] \)

\[
\begin{align*}
\sum_{i=0}^{m} & \left\{ \left[ 2 \ (x_1 - a_i) \ - x_4 - \sqrt{\left( (x_1 - a_i)^2 + (x_2 - b_i)^2 + (x_3 - c_i)^2 \right)} \right] + \{2, 2, 2, \} \right\} / \\
\left( \sqrt{\left( (x_1 - a_i)^2 + (x_2 - b_i)^2 + (x_3 - c_i)^2 \right)} \right) , \\
\sum_{i=0}^{m} & \left\{ \left[ 2 \ (x_2 - b_i) \ - x_4 - \sqrt{\left( (x_1 - a_i)^2 + (x_2 - b_i)^2 + (x_3 - c_i)^2 \right)} \right] + \{2, 2, 2, \} \right\} / \\
\left( \sqrt{\left( (x_1 - a_i)^2 + (x_2 - b_i)^2 + (x_3 - c_i)^2 \right)} \right) , \\
\sum_{i=0}^{m} & \left\{ \left[ 2 \ (x_3 - c_i) \ - x_4 - \sqrt{\left( (x_1 - a_i)^2 + (x_2 - b_i)^2 + (x_3 - c_i)^2 \right)} \right] + \{2, 2, 2, \} \right\} / \\
\left( \sqrt{\left( (x_1 - a_i)^2 + (x_2 - b_i)^2 + (x_3 - c_i)^2 \right)} \right) , \\
\sum_{i=0}^{m} & -2 \left\{ - x_4 - \sqrt{\left( (x_1 - a_i)^2 + (x_2 - b_i)^2 + (x_3 - c_i)^2 \right)} \right\} + \{2, 2, 2, \} \\
\end{align*}
\]

In our case \( m = 6 \), therefore the numeric form of the equations,

\( v_n = v / \ . \, m \to m / \ . \, \text{data} \, // \, \text{Expand}; \)

For example the first equation
\[v_n[[1]]\]
\[-1.05849 \times 10^8 \times 12 \times 1 + 6.80191 \times 10^{14} / \left( \sqrt{\left( \left( -1.50972 \times 10^7 \times x_1 \right)^2 + \left( 4.63609 \times 10^6 \times x_2 \right)^2 + \left( -2.13267 \times 10^7 \times x_3 \right)^2 \right)} \right] - \left( 4.50541 \times 10^7 \times x_1 \right) / \left( \sqrt{\left( \left( -1.50972 \times 10^7 \times x_1 \right)^2 + \left( 4.63609 \times 10^6 \times x_2 \right)^2 + \left( -2.13267 \times 10^7 \times x_3 \right)^2 \right)} \right) + 5.64346 \times 10^{13} / \left( \sqrt{\left( -1.39999 \times 10^6 \times x_1 \right)^2 + \left( 1.75637 \times 10^7 \times x_2 \right)^2 + \left( -1.97056 \times 10^7 \times x_3 \right)^2 \right)} \right] - \left( 4.03108 \times 10^7 \times x_1 \right) / \left( \sqrt{\left( -1.39999 \times 10^6 \times x_1 \right)^2 + \left( 1.75637 \times 10^7 \times x_2 \right)^2 + \left( -1.97056 \times 10^7 \times x_3 \right)^2 \right)} \right)
\]
\[3.43879 \times 10^{14} / \left( \sqrt{\left( -1.41776 \times 10^7 \times x_1 \right)^2 + \left( 1.88148 \times 10^7 \times x_2 \right)^2 + \left( -1.22439 \times 10^7 \times x_3 \right)^2 \right)} \right] + 1.1081 \times 10^{15} / \left( \sqrt{\left( -2.34603 \times 10^7 \times x_1 \right)^2 + \left( 9.43352 \times 10^6 \times x_2 \right)^2 + \left( -8.17494 \times 10^6 \times x_3 \right)^2 \right)} \right] - \left( 4.73483 \times 10^7 \times x_1 \right) / \left( \sqrt{\left( -2.34603 \times 10^7 \times x_1 \right)^2 + \left( 9.43352 \times 10^6 \times x_2 \right)^2 + \left( -8.17494 \times 10^6 \times x_3 \right)^2 \right)} \right] + 3.38999 \times 10^{14} / \left( \sqrt{\left( -6.99566 \times 10^6 \times x_1 \right)^2 + \left( 2.35378 \times 10^7 \times x_2 \right)^2 + \left( 9.92791 \times 10^6 \times x_3 \right)^2 \right)} \right] - \left( 4.84442 \times 10^7 \times x_1 \right) / \left( \sqrt{\left( -6.99566 \times 10^6 \times x_1 \right)^2 + \left( 2.35378 \times 10^7 \times x_2 \right)^2 + \left( 9.92791 \times 10^6 \times x_3 \right)^2 \right)} \right] - \left( 3.01944 \times 10^7 \times x_4 \right) / \left( \sqrt{\left( -1.50972 \times 10^7 \times x_1 \right)^2 + \left( 4.63609 \times 10^6 \times x_2 \right)^2 + \left( -2.13267 \times 10^7 \times x_3 \right)^2 \right)} \right] + (2 \times x_1 \times 4) / \left( \sqrt{\left( -1.50972 \times 10^7 \times x_1 \right)^2 + \left( 4.63609 \times 10^6 \times x_2 \right)^2 + \left( -2.13267 \times 10^7 \times x_3 \right)^2 \right)} \right] - \left( 2.79998 \times 10^8 \times x_4 \right) / \left( \sqrt{\left( -1.39999 \times 10^6 \times x_1 \right)^2 + \left( 1.75637 \times 10^7 \times x_2 \right)^2 + \left( -1.97056 \times 10^7 \times x_3 \right)^2 \right)} \right] + (2 \times x_1 \times 4) / \left( \sqrt{\left( -1.39999 \times 10^6 \times x_1 \right)^2 + \left( 1.75637 \times 10^7 \times x_2 \right)^2 + \left( -1.97056 \times 10^7 \times x_3 \right)^2 \right)} \right] + (1.6413 \times 10^7 \times x_4) / \left( \sqrt{\left( -1.820649 \times 10^6 \times x_1 \right)^2 + \left( 1.8218 \times 10^7 \times x_2 \right)^2 + \left( -1.76052 \times 10^7 \times x_3 \right)^2 \right)} \right] + (2 \times x_1 \times 4) / \left( \sqrt{\left( -1.820649 \times 10^6 \times x_1 \right)^2 + \left( 1.8218 \times 10^7 \times x_2 \right)^2 + \left( -1.76052 \times 10^7 \times x_3 \right)^2 \right)} \right] - (2.83551 \times 10^7 \times x_4) / \left( \sqrt{\left( -6.99176 \times 10^7 \times x_1 \right)^2 + \left( 1.88148 \times 10^7 \times x_2 \right)^2 + \left( -1.22439 \times 10^7 \times x_3 \right)^2 \right)} \right] + (2 \times x_1 \times 4) / \left( \sqrt{\left( -6.99176 \times 10^7 \times x_1 \right)^2 + \left( 1.88148 \times 10^7 \times x_2 \right)^2 + \left( -1.22439 \times 10^7 \times x_3 \right)^2 \right)} \right] - (4.69207 \times 10^7 \times x_4) / \left( \sqrt{\left( -2.34603 \times 10^7 \times x_1 \right)^2 + \left( 9.43352 \times 10^6 \times x_2 \right)^2 + \left( -8.17494 \times 10^6 \times x_3 \right)^2 \right)} \right] + (2 \times x_1 \times 4) / \left( \sqrt{\left( -2.34603 \times 10^7 \times x_1 \right)^2 + \left( 9.43352 \times 10^6 \times x_2 \right)^2 + \left( -8.17494 \times 10^6 \times x_3 \right)^2 \right)} \right] - (1.39913 \times 10^7 \times x_4) / \left( \sqrt{\left( -6.99566 \times 10^6 \times x_1 \right)^2 + \left( 2.35378 \times 10^7 \times x_2 \right)^2 + \left( 9.92791 \times 10^6 \times x_3 \right)^2 \right)} \right] + (2 \times x_1 \times 4) / \left( \sqrt{\left( -6.99566 \times 10^6 \times x_1 \right)^2 + \left( 2.35378 \times 10^7 \times x_2 \right)^2 + \left( 9.92791 \times 10^6 \times x_3 \right)^2 \right)} \right)

We pick up a fairly wrong initial guess provided by the solution of a triplet subset.

\[x_0 = \{ (596.951.52753, -4.8527795710 \times 10^6, 4.08875864269 \times 10^6, 3510.4002370764) \} \]

\{\{596.952., -4.85278 \times 10^6, 4.08876 \times 10^6, 3510.4\}\} \]

The variables,

\[v = \{ x_1, x_2, x_3, x_4 \}; \]
The start system itself - fixed point homotopy -,

\[ gF = V - \text{First}(X0) \]
\[ \{-596952. + x1, 4.85278 \times 10^6 + x2, -4.08876 \times 10^6 + x3, -3510.4 + x4\} \]

To avoid singularity of the homotopy function, let
\[ \gamma = \{1, 1, 1, 1\}; \]

\[
\text{AbsoluteTiming}[\text{solH} = \text{LinearHomotopyFR}(\text{vn}, gF, V, X0, \gamma, 10);]
\]
\[ \{0.133271, \text{Null}\} \]

\[
\text{solH}[[1]]
\]
\[ \{\{596.930, -4.84785 \times 10^6, 4.08823 \times 10^6, 15.5181\}\} \]
\[
\text{NumberForm}[%, 15]
\]
\[ \{\{596929.653490663, -4.84785155260446 \times 10^6, 4.08822679566195 \times 10^6, 15.5180507307228\}\} \]

Displaying the homotopy paths,
\[
\text{paH} = \text{PathsV}(V, \text{solH}[[2]], X0);
\]
\[
\text{paH} = \text{Flatten}[\text{paH}]
\]
Now we try to use nonlinear homotopy

```
AbsoluteTiming[
solH = NonLinearHomotopyFR[vn, gF, V, X0, γ, 3];
]
```

```
{0.0831006, Null}
solH[[1]]
```

```
{[596.930, -4.84785 \times 10^6, 4.08823 \times 10^6, 15.5181]}
```

Displaying the homotopy paths,
Now the running time of the nonlinear homotopy is shorter.

**Conclusions**

As demonstrated by these examples, the linear and nonlinear homotopy methods prove to be powerful solution tool in solving nonlinear geodetic problems, especially if it is difficult to find proper initial values to ensure convergence of local solution methods. These global numerical methods can be successful when symbolic computation based on Groebner basis or Dixon resultant fail because of the size and complexity of the problem. Since the
computations along the different paths are independent, parallel computation can be successfully employed. In general, nonlinear homotopy gives more precise result or required less iteration steps and needs less computation time at a fixed precision than its the linear counterpart.

References


