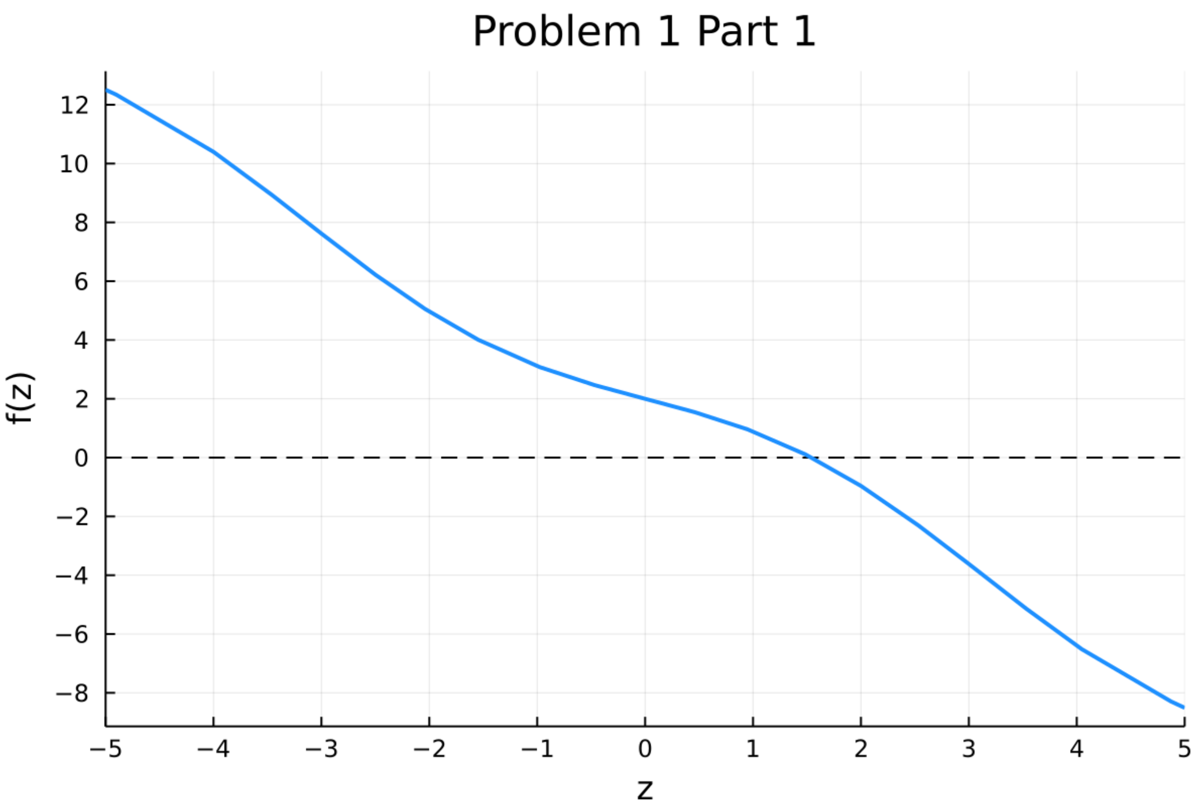
## Problem 1: Root Solving

**Problem Statement** Use the following equation, evaluated at $\mathbf{x} = [1,\ 2,\ 3]^T$, to complete parts 1-4:

$$F(\mathbf{x}, z) = 2z\sin(x_1 x_3 + x_2) + \sin(z)\cos(x_1 + x_2 + x_3) + 2$$

### Part 1.1

**Problem Statement** Plot the function, $F$, for $z$ values from -5 to 5.




### Part 1.2

**Problem Statement** Root solve the equation for $z$, using any method starting with a guess of $z = 0$. What is the root to 16 digits?

Using the Julia root solving package, `Roots.jl`, $F(z) = 0$ at the value of $\boxed{z = 1.543295599106779}$.

### Part 1.3

**Problem Statement** Take one correction step using a guess of $z_0 = 1.5$. Report the error of hte estimated root compared to the actual root when using:

1) First order recursive correction
2) Second order recursive correction
3) Third order recursive correction
4) Quadradic equation correction
5) Halley correction
6) Laguerre correction with an $n = 1$
7) Laguerre correction with an $n = 2$
8) Laguerre correction with an $n = 3$

Error from value obtained in Part 1.2 after one step:

| Method | Error |
|---|---|
| First Order Recursive Error | $4.8566623763979244e-4$ |
| Second Order Recursive Error | $1.0526718867920337e-5$ |
| Third Order Recursive Error | $2.069224032119621e-7$ |
| Quadratic Error | $4.1154893160033623e-7$ |
| Halley Error | $4.966159171448936e-6$ |
| Laguerre Error $n = 1$ | $4.8566623763979244e-4$ |
| Laguerre Error $n = 2$ | $4.1154893160033623e-7$ |
| Laguerre Error $n = 3$ | $9.107433529553788e-7$ |

### Part 1.4

**Problem Statement** Complete the root solve until convergence, to 16 digits, for each of the cases in Part 1.3, for two cases, $z_0 = 1.5$ and $z_0 = 0$. In tabular form, report the error at each iteration and discuss the results. State how you define convergence.

When solving for the roots of the function, the stop condition/convergence of the problem was defined as when the next step in the iteration goes below machine precision (indicated by the first zero for each column).

**Error per iteration per algorithm for $z_0 = 1.5$**

| Iter | halley | quad | lag1 | lag2 | lag3 | rec1 | rec2 | rec3 |
|---|---|---|---|---|---|---|---|---|
| 1 | -4.9661e-6 | 4.1154e-7 | 0.0004 | 4.1154e-7 | -9.1074e-7 | 0.0004 | -1.05267e-5 | 2.0692e-7 |
| 2 | 0.0 | 0.0 | 5.9831e-8 | 0.0 | 0.0 | 5.9831e-8 | -2.2203e-16 | 0.0 |
| 3 | 0.0 | 0 | 8.8817e-16 | 0.0 | 0.0 | 8.8817e-16 | 0.0 | 0.0 |

**Error per iteration per algorithm for $z_0 = 0.0$**

| Iter | halley | quad | lag1 | lag2 | lag3 | rec1 | rec2 | rec3 |
|---|---|---|---|---|---|---|---|---|
| 1 | 0.5450 | 0.5450 | 0.5450 | 0.5450 | 0.5450 | 0.5450 | 0.5450 | -0.9768 |
| 2 | 0.0101 | 0.0049 | 0.0558 | 0.0049 | 0.0064 | 0.0558 | 0.01407 | -0.1033 |
| 3 | 6.5701e-8 | -2.4337e-10 | 0.0007 | -2.4343e-10 | 3.8439e-9 | 0.0007 | 3.4951e-7 | 6.8823e-6 |
| 4 | 0.0 | 2.2204e-16 | 1.4989e-7 | 0.0 | 0.0 | 1.4989e-7 | 0.0 | 0.0 |
| 5 | 0 | 0.0 | 5.9952e-15 | 0 | 0 | 5.9952e-15 | 0 | 0 |

All methods were able to reach the root to within machine precision. Solvers that used lower order derivatives, on average, took longer to solve for each initial position. Additionally, for the $z_0 = 0$ case, the error between the actual root and the position after the first step was the same for all terms, except for the third order recursive solver. I attribute this to the fact that the third order recursive solver is the only one to use a third derivative.

### Problem 2: Derivatives of a Root Solve Process

**Problem Statement** Use the following equation, evaluated at $\mathbf{x} = [1,\ 2,\ 3]^T$, to complete parts 1-4:

$$P(\mathbf{x}, z) = \cos\left(-x_1 x_2 x_3 + z^2\right)$$

#### Part 2.1

**Problem Statement** Using the complex step method, compute the total derivative of $P$ with respect to $\mathbf{x}$.

Evaluating the root solver as a black box, the following derivatives are evaluated by the complex step at the $z_0$ values from Part 1.

$$\left.\frac{dP}{d\mathbf{x}}\right|_{z_0=z^\star} = \begin{bmatrix} 0.5771 \\ 0.5118 \\ 0.0530 \end{bmatrix}$$

#### Part 2.2

**Problem Statement** Using the analytical method method, compare the total derivative of $P$ with respect to $\mathbf{x}$.

Using the following equation for the total derivative of $P$ with respect to the vector $\mathbf{x}$, $\frac{dP}{d\mathbf{x}} = \frac{\partial P}{\partial \mathbf{x}} + \frac{\partial P}{\partial z}\frac{dz}{d\mathbf{x}}$ where $\frac{dz}{d\mathbf{x}} = -\frac{F_\mathbf{x}}{F_z}$. MATLAB's symbolic toolbox was used to take the partial derivatives of $P$ and $F$ with respect to each variable, which evaluates to:

$$\frac{dP}{d\mathbf{x}} = \begin{bmatrix} x_2\, x_3\, \sigma_2 - \frac{2\, z\, \sigma_2\, (\sigma_3 - 2\, x_3\, z\, \cos(x_2 + x_1\, x_3))}{\sigma_1} \\ x_1\, x_3\, \sigma_2 + \frac{2\, z\, \sigma_2\, (2\, z\, \cos(x_2 + x_1\, x_3) - \sigma_3)}{\sigma_1} \\ x_1\, x_2\, \sigma_2 - \frac{2\, z\, \sigma_2\, (\sigma_3 - 2\, x_1\, z\, \cos(x_2 + x_1\, x_3))}{\sigma_1} \end{bmatrix}$$

where

$$\sigma_1 = 2\, \sin(x_2 + x_1\, x_3) + \cos(x_1 + x_2 + x_3)\, \cos(z)$$
$$\sigma_2 = \sin(z^2 - x_1\, x_2\, x_3)$$
$$\sigma_3 = \sin(x_1 + x_2 + x_3)\, \sin(z)$$

Evaluating at $z^\star$, the resulting gradient is $\frac{dP}{d\mathbf{x}} = [0.5771,\, 0.5118,\, 0.0530]^T$. Error between analytical and complex methods comes out to:

$$\varepsilon = [-0.4441,\, 0,\, -0.2220]^T \times 10^{-15}$$

**Part 2.3**

**Problem Statement** Using the complex step method, compute the second total derivative of $P$ with respect to $\mathbf{x}$ using the equations found in Part 2.2 and the value for $F(z) = 0$ found in Part 1.2.

$$\left.\frac{d^2P}{d\mathbf{x}^2}\right|_{z_0 = z^\star} = \begin{bmatrix} -22.2226 & -4.9360 & -7.1680 \\ -4.9360 & -0.9918 & -1.5241 \\ -7.1680 & -1.5241 & -2.0855 \end{bmatrix}$$

**Part 2.4**

**Problem Statement** Using the analytical method, compute the second total derivative of $P$ with respect to $\mathbf{x}$. Describe the approach. Compare answer to Part 2.3. Also report the Frobenius norm.

Once again using MATLAB to calculate intermediate partials $F_{\mathbf{xx}}$, $F_{\mathbf{x}z}$, $F_{zz}$, and $F_{z\mathbf{x}}$, as well as $P_{\mathbf{xx}}$, $P_{\mathbf{x}z}$, $P_{zz}$, and $P_{z\mathbf{x}}$. The second total derivative of $z$ with respect to $\mathbf{x}$ was then calculated using the above terms:

$$\frac{d^2z}{d\mathbf{x}^2} = F_z^{-2} F_{\mathbf{x}}^T (F_{z\mathbf{x}} + F_{zz}\frac{dz}{d\mathbf{x}}) - F_z^{-1}(F_{\mathbf{x}\mathbf{x}} + F_{\mathbf{x}z}\frac{dz}{d\mathbf{x}})$$

Taking this term, and all previous, can then be transformed into the second total derivative of $P$ with respect to **x**.

$$\frac{d^2P}{d\mathbf{x}^2} = P_{\mathbf{x}\mathbf{x}} + (P_{\mathbf{x}z}\frac{dz}{d\mathbf{x}} + \frac{dz}{d\mathbf{x}}^T P_{z\mathbf{x}}) + P_{zz}\frac{dz}{d\mathbf{x}}^T\frac{dz}{d\mathbf{x}} + P_z\frac{d^2z}{d\mathbf{x}^2}$$

The total expression is too large to fit within one page, so it will be in Appendix A. When evaluating the expression for $F(z) = 0$, the following Hessian is found:

$$\left.\frac{d^2P}{d\mathbf{x}^2}\right|_{z_0=z^\star} = \begin{bmatrix} -14.4862 & -1.8615 & -4.0935 \\ 0.6120 & 1.2130 & 0.6807 \\ -5.8336 & -0.9938 & -1.5551 \end{bmatrix}$$

These are different from the values found by the complex step. After some experimentation, the values of each method **do** match up where $z = 0$. This leads me to believe my second parital derivatives with respect to $z$ are incorrect, most likely due to linear algebra operations or typos. See Appendix B for those values.

Consequently, both matrix norms came out to different values. The complex step method reported a Frobenius norm of $\boxed{16.4261}$, while the analytical method reported a norm of $\boxed{25.5999}$.

## Problem 3: Chain Rule for Orbit Problems

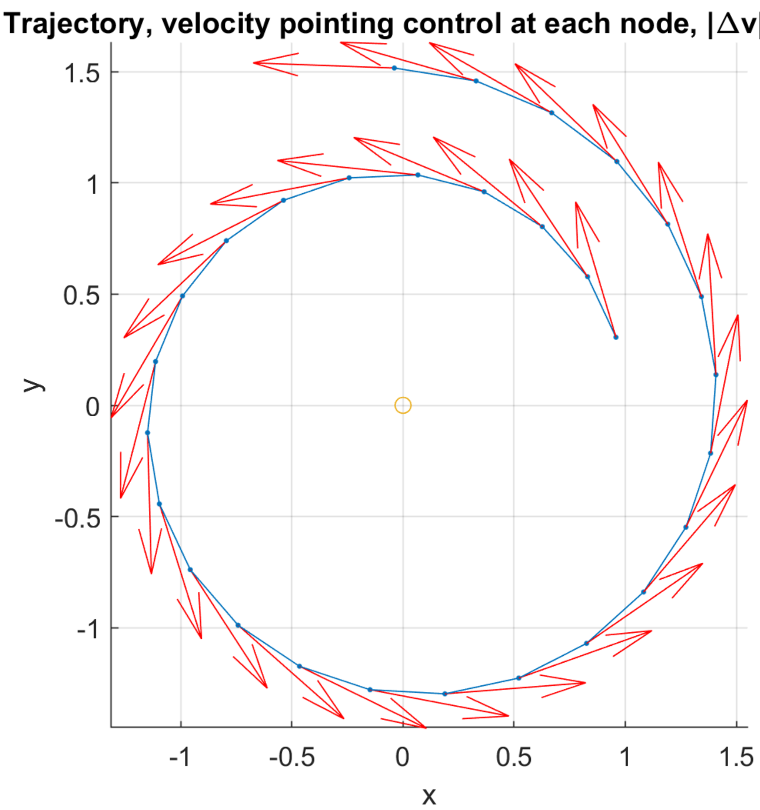
**Problem Statement** Use the Kepler UV propagator from Canvas for all parts.

### Part 3.1

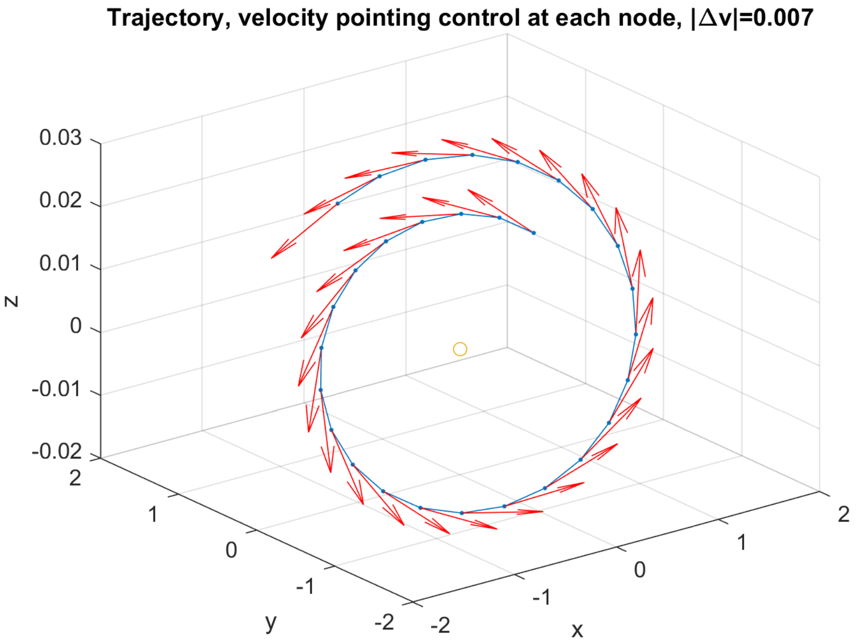
**Problem Statement** Implement the orbit simulation using the following characteristics:

- A starting state of $\mathbf{x}_0 = [1,\ 0.01,\ 0.01,\ 0.01,\ 1,\ 0.01]$, and a standard gravitational parameter of $\mu = 1$
- Number of nodes, $n$, equal to 30, each traversing a change in universal variable of $\Delta E = 0.3$, with a $\Delta\mathbf{v} = [7 \times 10^{-3},\ 0,\ 0]$ at the end of each leg.
- Computes a performance index at the end of the complete trajectory using the cost function: $J = (\mathbf{r}_f - \mathbf{r}_\star)^T(\mathbf{r}_f - \mathbf{r}_\star)$, where $\mathbf{r}_\star = [3,\ 0,\ 0.1]$, and $\mathbf{r}$ is the first 3 components of $\mathbf{x}$

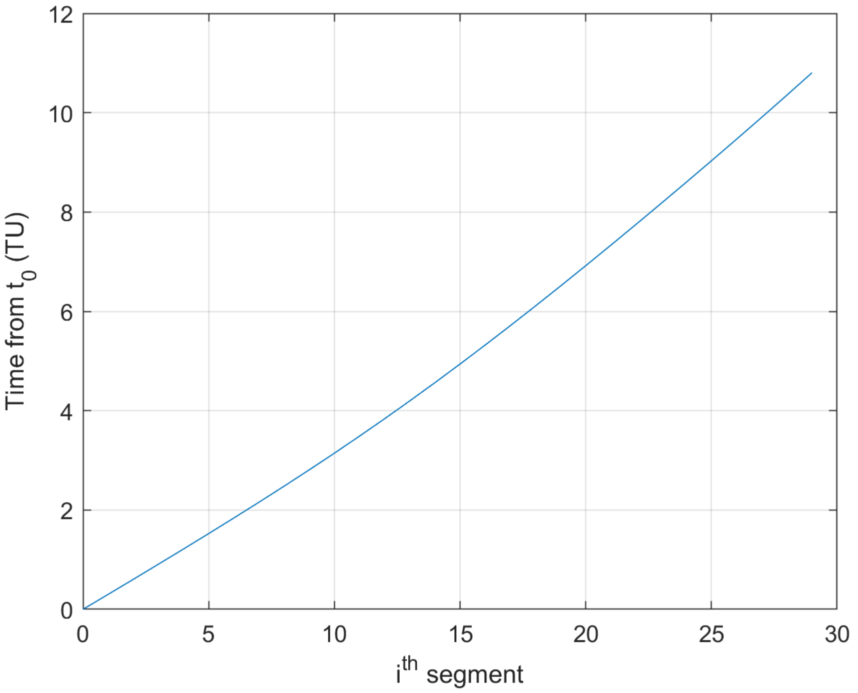
**Part 3.1.1:** Plot top view of trajectory



**Part 3.1.2:** Plot 3D view of trajectory




**Part 3.1.3:** Plot current time as a function of the node




**Part 3.1.4:** State the value of J

When evaluating the performance index $J$ for this trajectory, found the value of $\boxed{11.5459}$ with an $\vec{r}_f = [0,\ 0,\ 0]$.

**Part 3.2**

**Problem Statement** Compute the partials of J with respect to **x**_0 using the chain rule and the STMs that result from the provided analytical propagator.

**Part 3.2.1:** Discuss and show the quations how you compute $\frac{dJ}{d\mathbf{x}_0}$.

**Part 3.2.2:** Turn in numeric values for $\frac{dJ}{d\mathbf{x}_0}$. Report the frobenius norm.

**Part 3.2.3:** Verify your partials are correct with finite differencing or complex step.

**Part 3.2.4:** Plot the function $\frac{dJ}{dx_{1i}}$ where $x_1$ is the first component of **x** and $i$ indicates the value at the beginning of the $i^{\text{th}}$ leg. Give equations used to compute values.

## Appendicies

### Appendix A Second Total Derivative

The $3 \times 3$ matrix was transformed into a vector for better visibility. Rows 1-3 are the first column, 4-6 is the second, etc. $\frac{d^2 P}{d\mathbf{x}^2} =$

$$\begin{pmatrix} \frac{4\,x_2\,x_3\,z\,\cos(\sigma_{25})\,\sigma_{24}}{\sigma_{23}} - \frac{{\sigma_{24}}^2\,\sigma_{20}}{{\sigma_{23}}^2} - 2\,z\,\sin\left(\sigma_{25}\right)\,\left(\frac{\sigma_{27}\,\sin(z) + 2\,{x_3}^2\,z\,\sigma_{26} - \frac{\sigma_4\,\sigma_{24}}{\sigma_{23}}}{\sigma_{23}} + \frac{\sigma_{24}\,\sigma_1}{{\sigma_{23}}^2}\right) - {x_2}^2\,{x_3}^2\,\cos\left(\sigma_{25}\right) \\ x_3\,\sin\left(\sigma_{25}\right) - 2\,z\,\sin\left(\sigma_{25}\right)\,\left(\frac{\sigma_{27}\,\sin(z) + \frac{\sigma_4\,\sigma_{22}}{\sigma_{23}} + 2\,x_3\,z\,\sigma_{26}}{\sigma_{23}} - \frac{\sigma_{24}\,\sigma_3}{{\sigma_{23}}^2}\right) + \sigma_7 - \sigma_{17} + \sigma_{13} - \sigma_9 \\ x_2\,\sin\left(\sigma_{25}\right) + 2\,z\,\sin\left(\sigma_{25}\right)\,\left(\frac{2\,z\,\sigma_{28} - \sigma_{27}\,\sin(z) + \frac{\sigma_4\,\sigma_{21}}{\sigma_{23}} - \sigma_{19}}{\sigma_{23}} - \frac{\sigma_{24}\,\sigma_2}{{\sigma_{23}}^2}\right) - \sigma_{16} - \sigma_8 + \sigma_{14} + \sigma_{10} \\ x_3\,\sin\left(\sigma_{25}\right) - 2\,z\,\sin\left(\sigma_{25}\right)\,\left(\frac{\sigma_{27}\,\sin(z) - \frac{\sigma_{18}\,\sigma_{24}}{\sigma_{23}} + 2\,x_3\,z\,\sigma_{26}}{\sigma_{23}} - \frac{\sigma_{22}\,\sigma_1}{{\sigma_{23}}^2}\right) + \sigma_7 - \sigma_{17} + \sigma_{13} - \sigma_9 \\ -{x_1}^2\,{x_3}^2\,\cos\left(\sigma_{25}\right) - 2\,z\,\sin\left(\sigma_{25}\right)\,\left(\frac{2\,z\,\sigma_{26} + \sigma_{27}\,\sin(z) + \frac{\sigma_{22}\,\sigma_{18}}{\sigma_{23}}}{\sigma_{23}} + \frac{\sigma_{22}\,\sigma_3}{{\sigma_{23}}^2}\right) - \frac{{\sigma_{22}}^2\,\sigma_{20}}{{\sigma_{23}}^2} - \frac{4\,x_1\,x_3\,z\,\cos(\sigma_{25})\,\sigma_{22}}{\sigma_{23}} \\ x_1\,\sin\left(\sigma_{25}\right) - 2\,z\,\sin\left(\sigma_{25}\right)\,\left(\frac{\sigma_{27}\,\sin(z) - \frac{\sigma_{18}\,\sigma_{21}}{\sigma_{23}} + 2\,x_1\,z\,\sigma_{26}}{\sigma_{23}} - \frac{\sigma_{22}\,\sigma_2}{{\sigma_{23}}^2}\right) + \sigma_6 - \sigma_{15} + \sigma_{11} - \sigma_{12} \\ x_2\,\sin\left(\sigma_{25}\right) + 2\,z\,\sin\left(\sigma_{25}\right)\,\left(\frac{2\,z\,\sigma_{28} - \sigma_{27}\,\sin(z) + \frac{\sigma_5\,\sigma_{24}}{\sigma_{23}} - \sigma_{19}}{\sigma_{23}} - \frac{\sigma_{21}\,\sigma_1}{{\sigma_{23}}^2}\right) - \sigma_{16} - \sigma_8 + \sigma_{14} + \sigma_{10} \\ x_1\,\sin\left(\sigma_{25}\right) - 2\,z\,\sin\left(\sigma_{25}\right)\,\left(\frac{\sigma_{27}\,\sin(z) + \frac{\sigma_5\,\sigma_{22}}{\sigma_{23}} + 2\,x_1\,z\,\sigma_{26}}{\sigma_{23}} - \frac{\sigma_{21}\,\sigma_3}{{\sigma_{23}}^2}\right) + \sigma_6 - \sigma_{15} + \sigma_{11} - \sigma_{12} \\ \frac{4\,x_1\,x_2\,z\,\cos(\sigma_{25})\,\sigma_{21}}{\sigma_{23}} - \frac{{\sigma_{21}}^2\,\sigma_{20}}{{\sigma_{23}}^2} - 2\,z\,\sin\left(\sigma_{25}\right)\,\left(\frac{\sigma_{27}\,\sin(z) + 2\,{x_1}^2\,z\,\sigma_{26} - \frac{\sigma_5\,\sigma_{21}}{\sigma_{23}}}{\sigma_{23}} + \frac{\sigma_{21}\,\sigma_2}{{\sigma_{23}}^2}\right) - {x_1}^2\,{x_2}^2\,\cos\left(\sigma_{25}\right) \end{pmatrix}$$

where

$$\sigma_1 = \sigma_{29}\,\cos\left(z\right) - 2\,x_3\,\sigma_{28} + \frac{\sigma_{27}\,\sin(z)\,\sigma_{24}}{\sigma_{23}}$$

$$\sigma_2 = \sigma_{29}\,\cos\left(z\right) - 2\,x_1\,\sigma_{28} + \frac{\sigma_{27}\,\sin(z)\,\sigma_{21}}{\sigma_{23}}$$

$$\sigma_3 = 2\,\sigma_{28} - \sigma_{29}\,\cos\left(z\right) + \frac{\sigma_{27}\,\sin(z)\,\sigma_{22}}{\sigma_{23}}$$

$$\sigma_4 = 2\,x_3\,\sigma_{28} - \sigma_{29}\,\cos\left(z\right)$$

$$\sigma_5 = 2\,x_1\,\sigma_{28} - \sigma_{29}\,\cos\left(z\right)$$

$$\sigma_6 = \frac{\sigma_{22}\,\sigma_{21}\,\sigma_{20}}{{\sigma_{23}}^2}$$

$$\sigma_7 = \frac{\sigma_{22}\,\sigma_{24}\,\sigma_{20}}{{\sigma_{23}}^2}$$

$$\sigma_8 = \frac{\sigma_{21}\,\sigma_{24}\,\sigma_{20}}{{\sigma_{23}}^2}$$

$$\sigma_9 = \frac{2\,x_2\,x_3\,z\,\cos(\sigma_{25})\,\sigma_{22}}{\sigma_{23}}$$

$$\sigma_{10} = \frac{2\,x_2\,x_3\,z\,\cos(\sigma_{25})\,\sigma_{21}}{\sigma_{23}}$$

$$\sigma_{11} = \frac{2\,x_1\,x_3\,z\,\cos(\sigma_{25})\,\sigma_{21}}{\sigma_{23}}$$

$$\sigma_{12} = \frac{2\,x_1\,x_2\,z\,\cos(\sigma_{25})\,\sigma_{22}}{\sigma_{23}}$$

$$\sigma_{13} = \frac{2\,x_1\,x_3\,z\,\cos(\sigma_{25})\,\sigma_{24}}{\sigma_{23}}$$

$$\sigma_{14} = \frac{2\,x_1\,x_2\,z\,\cos(\sigma_{25})\,\sigma_{24}}{\sigma_{23}}$$

$$\sigma_{15} = {x_1}^2\,x_2\,x_3\,\cos\left(\sigma_{25}\right)$$

$$\sigma_{16} = x_1\,{x_2}^2\,x_3\,\cos\left(\sigma_{25}\right)$$

$$\sigma_{17} = x_1\,x_2\,{x_3}^2\,\cos\left(\sigma_{25}\right)$$

$$\sigma_{18} = 2\,\sigma_{28} - \sigma_{29}\,\cos\left(z\right)$$

$$\sigma_{19} = 2\,x_1\,x_3\,z\,\sigma_{26}$$

$$\sigma_{20} = 2\,\sin\left(\sigma_{25}\right) + 4\,z^2\,\cos\left(\sigma_{25}\right)$$

$$\sigma_{21} = \sigma_{29}\,\sin\left(z\right) - 2\,x_1\,z\,\sigma_{28}$$

$$\sigma_{22} = 2\,z\,\sigma_{28} - \sigma_{29}\,\sin\left(z\right)$$

$$\sigma_{23} = 2\,\sigma_{26} + \sigma_{27}\,\cos\left(z\right)$$

$$\sigma_{24} = \sigma_{29}\,\sin\left(z\right) - 2\,x_3\,z\,\sigma_{28}$$

$$\sigma_{25} = z^2 - x_1\,x_2\,x_3$$

$$\sigma_{26} = \sin\left(x_2 + x_1\,x_3\right)$$

$$\sigma_{27} = \cos\left(x_1 + x_2 + x_3\right)$$

$$\sigma_{28} = \cos\left(x_2 + x_1\,x_3\right)$$

$$\sigma_{29} = \sin\left(x_1 + x_2 + x_3\right)$$

## Appendix B Second Total Derivative Values

$$\left.\frac{d^2P}{d\mathbf{x}^2}\right|_{z_0=0}^{analytical} = \begin{bmatrix} -34.5661 & -16.4448 & -10.9632 \\ -16.4448 & -8.6415 & -5.4816 \\ -10.9632 & -5.4816 & -3.8407 \end{bmatrix}$$

$$\left.\frac{d^2P}{d\mathbf{x}^2}\right|_{z_0=0}^{CX} = \begin{bmatrix} -34.5661 & -16.4448 & -10.9632 \\ -16.4448 & -8.6415 & -5.4816 \\ -10.9632 & -5.4816 & -3.8407 \end{bmatrix}$$

```
Main.workspace2.OptimalSpacecraftTrajectories
```

```
begin
    using Plots ✓ , PlutoUI ✓ , LinearAlgebra ✓ , Symbolics ✓ ,
DifferentialEquations ✓ , DataFrames ✓ , CSV ✓
    include("/home/burtonyale/Documents/repos
/OptimalSpacecraftTrajectories/src/OptimalSpacecraftTrajectories.jl"
    import .OptimalSpacecraftTrajectories
    const OST = OptimalSpacecraftTrajectories
end
```

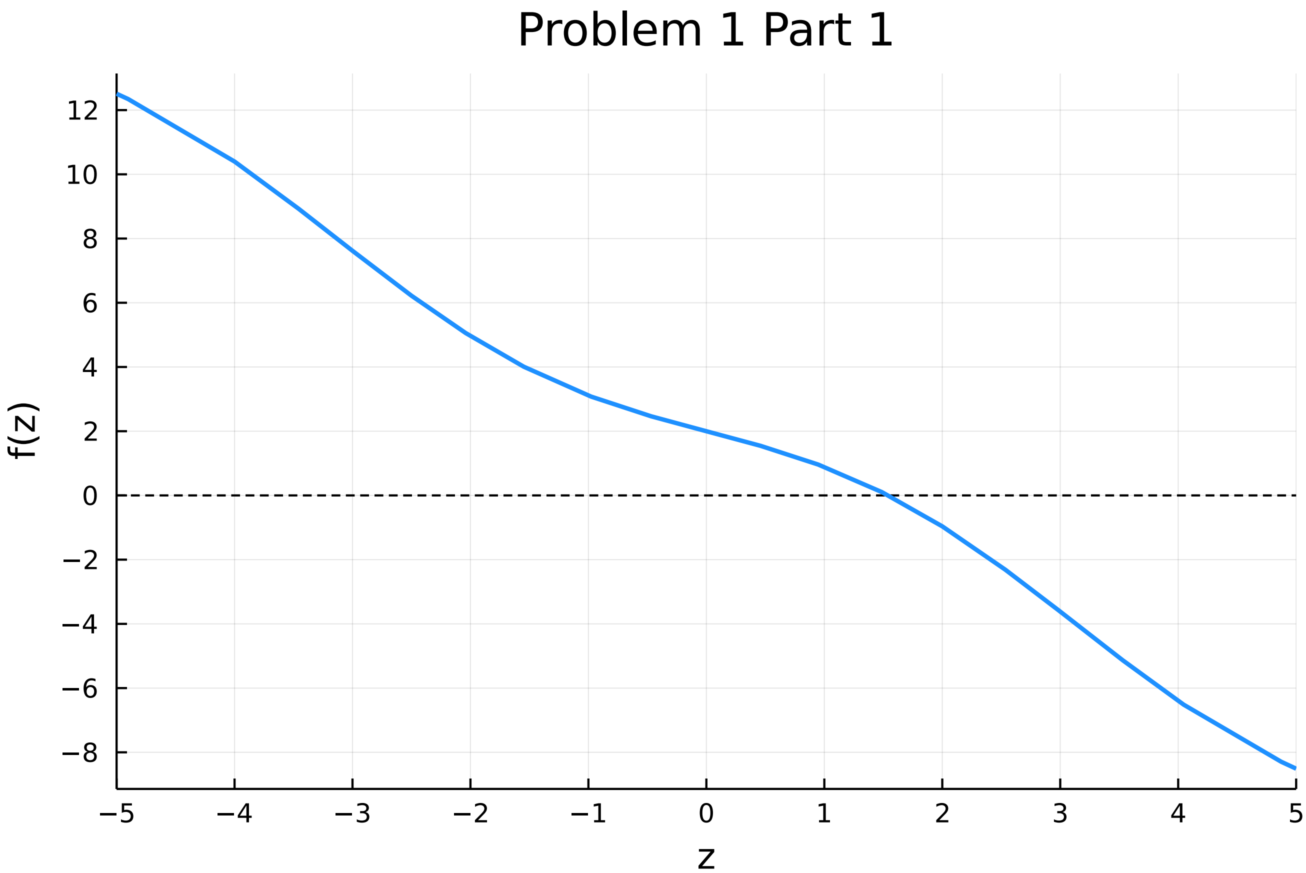
# Problem 1

## Part 1

```
begin
    # ORIGINAL FUNCTION
    F = (x⃗, z) -> 2*z*sin(x⃗[1]*x⃗[3] + x⃗[2]) + sin(z)*cos(x⃗[1] + x⃗[2]
x⃗[3]) + 2

    # SETTING X VAL
    x⃗₀ = [1.0, 2.0, 3.0]

    # X VAL SET
    Fz(z) = F(x⃗₀, z)
    md"## Part 1"
end
```




```
begin
    plot([-5, 5], [0, 0], linestyle=:dash, color=:black, label="",
xlim=(-5, 5), fmt=:png)
    plot!(Fz, -5, 5, color=:dodgerblue, lw=2, label="", dpi=200,
xticks=-5:5, xlabel="z", yticks=-10:2:14, ylabel="f(z)", title="Prob
Part 1")
    # png("hw2p1.1.png")
end
```

# Part 2

### f(z) = 0 @ z = 1.543295599106779

Using Roots.jl package

```
begin
    # δ, z = root_solve(ƒ_z, 0.0, 3, 1e-16)
    # plot([-5, 5], [0, 0], linestyle=:dash, color=:black, label="",
    #   title="Problem 1 Part 2: 3rd Order Recursive")
    # plot!(ƒ_z, -5, 5, color=:dodgerblue, lw=2,
    #   label="f(z) = 0 @ z = $(round(z[end], digits=6))", dpi=200,
    #   xticks=-5:5, xlabel="z", yticks=-10:2:14, ylabel="f(z)")
    # scatter!([z[1]], [ƒ_z(z[1])], marker=:star, color=:red,
label="Initial Guess")
    # scatter!(z[2:end], ƒ_z.(z[2:end]), color=:red, label="Number o
Iterations = $(length(δ))")
    using Roots ✓
    z_actual = find_zero(Fz, 0.0)
    md"""
    ### f(z) = 0 @ z = $(z_actual)

    Using Roots.jl package
    """
end
```

## Part 3

root_solve_recursive (generic function with 2 methods)

```
function root_solve_recursive(f, x*, order, minerror = 1e-3)
    # GENERATING SAVED VARIABLES
    Δz = [Inf]
    x_out = [x*]

    # ROOT SOLVING
    # while abs(Δz[end]) > minerror
        # FINDING DERIVATIVES
        f₀ = f(x*)
        f′ = OST.cntrDiff[5][1](f, x*, 1e-5)
        f″ = OST.cntrDiff[5][2](f, x*, 1e-3)
        f‴ = OST.cntrDiff[5][3](f, x*, 1e-3)

        # GENERATING TAYLOR SERIES
        δz = -f₀/f′
        δ²z = (-f″*f₀^2)/f′^3
        δ³z = (f₀^3 / f′^5)*(f′*f‴ - 3*f″^2)
        if order == 1
            Δz₀ = δz
        elseif order == 2
            Δz₀ = δz + 0.5*δ²z
        else
            Δz₀ = δz + 0.5*δ²z + 1/factorial(3) * δ³z
        end
        push!(Δz, Δz₀)
        x* += Δz₀
        push!(x_out, x*)
    # end
    deleteat!(Δz, 1)
    return Δz₀
end
```

root_solve_quad (generic function with 1 method)

```
function root_solve_quad(f, x*)
    # FINDING DERIVATIVES
    f₀ = f(x*)
    f′ = OST.cntrDiff[5][1](f, x*, 1e-5)
    f″ = OST.cntrDiff[5][2](f, x*, 1e-3)
    f‴ = OST.cntrDiff[5][3](f, x*, 1e-3)

    if f′^2 - 2*f″*f₀ < 0; return 0; end

    Δx = [(-f′ - sqrt(f′^2 - 2*f″*f₀))/f″, (-f′ + sqrt(f′^2 -
2*f″*f₀))/f″]

    if abs(f(x* + Δx[1])) < abs(f₀)
        return Δx[1]
    else
        return Δx[2]
    end

end
```

root_solve_halley (generic function with 1 method)

```
function root_solve_halley(f, x*)
    # FINDING DERIVATIVES
    f₀ = f(x*)
    f′ = OST.cntrDiff[5][1](f, x*, 1e-5)
    f″ = OST.cntrDiff[5][2](f, x*, 1e-3)
    f‴ = OST.cntrDiff[5][3](f, x*, 1e-3)

    Δx = (2*f₀*f′)/(f″*f₀ - 2*f′^2)

end
```

```
root_solve_laguerre (generic function with 1 method)
```

```julia
function root_solve_laguerre(f, x*, n)
    # FINDING DERIVATIVES
    f₀ = f(x*)
    f′ = OST.cntrDiff[5][1](f, x*, 1e-5)
    f″ = OST.cntrDiff[5][2](f, x*, 1e-3)
    f‴ = OST.cntrDiff[5][3](f, x*, 1e-3)

    D₁ = f′/f₀
    D₂ = f″/f₀ - D₁^2


    if (1-n) * (D₁^2 + D₂*n) < 0; return 0; end

    q = [-n/(D₁ - sqrt( (1-n) * (D₁^2 + D₂*n) )),
         -n/(D₁ + sqrt( (1-n) * (D₁^2 + D₂*n) ))]
    if abs(q[1]) < abs(q[2])
        return q[1]
    else
        return q[2]
    end
end
```

a) First Order Recursive Error: 0.00048566623763979244

b) Second Order Recursive Error: -1.0526718867920337e-5

c) Third Order Recursive Error: 2.0692240321196 21e-7

d) Quadratic Error: 4.1154893160033623e-7

e) Halley Error: -4.966159171448936e-6

f) Laguerre $n = 1$ Error: 0.00048566623763979244

g) Laguerre $n = 2$ Error: 4.1154893160033623e-7

h) Laguerre $n = 3$ Error: -9.107433529553788e-7

# Part 4

```
converge_root_solve (generic function with 1 method)
```

```julia
function converge_root_solve(f, x⋆, solver; output_type=:x,
min_convergence=1e-16)
    # SETUP
    Δx = []
    x = []

    # FIRST ITERATION
    push!(Δx, solver(f, x⋆))
    x⋆ += Δx[end]
    push!(x, x⋆)

    # SECOND ITERATION
    iters = 1
    # while (abs(x⋆ - z_actual) > min_convergence)
    while abs(Δx[end]) > min_convergence
        δx = solver(f, x⋆)
        if abs(real(δx)) < min_convergence || isnan(δx); break; end
        # if ((δx < min_convergence) & (Δx[end] < min_convergence))
isnan(δx); break; end
        push!(Δx, δx)
        x⋆ += Δx[end]
        push!(x, x⋆)
        iters += 1
        # if iters > 30
        #   break
        # end
    end

    while length(x) < 10
        push!(x, 0)
    end

    if output_type === :x
        return x⋆
    elseif output_type === :Δxx
        return Δx, x, x .- z_actual
    end
    # return x, Δx, x⋆
end
```

"hw2p1.4_0.csv"

```
begin
    solvers = [root_solve_halley,
        root_solve_quad,
        [laguerre(f, z) = root_solve_laguerre(f, z, n) for n = 1:3].
        [recursion(f, z) = root_solve_recursive(f, z, n) for n = 1:3
    outputs = []
    df = []
    for z₀ = [1.5, 0.0], i = 1:length(solvers)
        push!(outputs, converge_root_solve(Fz, z₀, solvers[i],
output_type=:Δxx, min_convergence=1e-16))
        push!(df, DataFrame(δz = outputs[end][2], z = outputs[end][2
err = outputs[end][3]))
        CSV.write("hw2_$i$(string(solvers[i]))$z₀.csv", df[end])
    end
    df
    CSV.write("hw2p1.4_0.csv", DataFrame(halley = df[9][!, 3], quad
df[10][!, 3], lag1 = df[11][!, 3], lag2 = df[12][!, 3], lag3 = df[13
3], rec1 = df[14][!, 3], rec2 = df[15][!, 3], rec3 = df[16][!, 3]))
end
```

# Problem 2

$$P = \cos\left(-x_1 x_2 x_3 + z^2\right) \qquad \mathbf{x} = [1,\, 2,\, 3]^T \qquad z_0 = 0$$

## Part 1

cplxDiff (generic function with 1 method)

```
function cplxDiff(f, x₀, h)
    f′ = []
    L = length(x₀)
    for i = 1:L
        H = zeros(Complex{Float64}, L)
        H[i] = h*im
        push!(f′, imag(f(x₀ + H))/h)
    end
    return f′
end
```

```
clear; clc
```

# Problem 2

## Part 1

```
funcp = @(x, z) cos(-x(1)*x(2)*x(3) + z^2);
funcF = @(x, z) 2*z*sin(x(1)*x(3) + x(2)) + sin(z)*cos(x(1) +x(3) + x(2)) + 2;
cplxDiff(@(x) funcp(x, 0), [1; 2; 3], 1e-20)
```

```
ans = 1×3
    1.6765    0.8382    0.5588
```

## Part 2

```
syms x_1 x_2 x_3 z
x = [x_1; x_2; x_3];
p = cos(-x(1)*x(2)*x(3) + z^2);
F = 2*z*sin(x(1)*x(3) + x(2)) + sin(z)*cos(x(1) + x(3) + x(2)) + 2
```

F = $2\,z\sin(x_2 + x_1\,x_3) + \cos(x_1 + x_2 + x_3)\sin(z) + 2$

```
Fx = [diff(F, x(1)) diff(F, x(2)) diff(F, x(3))];
Fz = diff(F, z);
dzdx = (-Fx/Fz)
```

dzdx =

$$\left(\frac{\sin(x_1 + x_2 + x_3)\sin(z) - 2\,x_3\,z\cos(x_2 + x_1\,x_3)}{\sigma_1} \quad -\frac{2\,z\cos(x_2 + x_1\,x_3) - \sin(x_1 + x_2 + x_3)\sin(z)}{\sigma_1} \quad \frac{\sin(x_1 + x}{}\right.$$

where

$$\sigma_1 = 2\sin(x_2 + x_1\,x_3) + \cos(x_1 + x_2 + x_3)\cos(z)$$

```
px = [diff(p, x(1)) diff(p, x(2)) diff(p, x(3))]
```

px = $\left(x_2\,x_3\sin\left(z^2 - x_1\,x_2\,x_3\right) \quad x_1\,x_3\sin\left(z^2 - x_1\,x_2\,x_3\right) \quad x_1\,x_2\sin\left(z^2 - x_1\,x_2\,x_3\right)\right)$

```
pz = diff(p, z)
```

pz = $-2\,z\sin\left(z^2 - x_1\,x_2\,x_3\right)$

```
dpdx = px + pz*dzdx
```

dpdx =

$$\left(x_2\,x_3\,\sigma_2 - \frac{2\,z\,\sigma_2\,\left(\sigma_3 - 2\,x_3\,z\cos(x_2 + x_1\,x_3)\right)}{\sigma_1} \quad x_1\,x_3\,\sigma_2 + \frac{2\,z\,\sigma_2\,\left(2\,z\cos(x_2 + x_1\,x_3) - \sigma_3\right)}{\sigma_1} \quad x_1\,x_2\,\sigma_2 - \frac{2\,z\,\sigma_2\,(}{}\right.$$

where

$$\sigma_1 = 2\sin(x_2 + x_1\,x_3) + \cos(x_1 + x_2 + x_3)\cos(z)$$

$$\sigma_2 = \sin\left(z^2 - x_1\,x_2\,x_3\right)$$

$$\sigma_3 = \sin(x_1 + x_2 + x_3)\sin(z)$$

```
func = matlabFunction(dpdx);
func_test = matlabFunction(dzdx);
% func(1, 2, 3, 0)
```

```
z_test = 1.543295599106779
```

```
z_test = 1.5433
```

```
testa = func(1, 2, 3, z_test)
```

```
testa = 1×3
    0.5771    0.5118    0.0530
```

```
tic; func(1,2,3,z_test); toc
```

```
Elapsed time is 0.000588 seconds.
```

```
% cplxDiff(@(x) funcp(x, z_test), [1; 2; 3], 1e-20) % Wrong way to do it
testb = aaa(funcF, funcp, [1,2,3], z_test)
```

```
testb = 1×3
    0.5771    0.5118    0.0530
```

```
tic; aaa(funcF, funcp, [1,2,3], z_test); toc
```

```
Elapsed time is 0.004930 seconds.
```

```
testa-testb
```

ans = 1×3
$10^{-15}$ ×

```
   -0.4441   -0.2220   -0.2220
```

## Part 3

```
cplxDiff(@(x) func(x(1), x(2), x(3), 0.0), [1;2;3], 1e-20)
```

```
ans = 3×3
  -34.5661  -16.4448  -10.9632
  -16.4448   -8.6415   -5.4816
  -10.9632   -5.4816   -3.8407
```

## Part 4

```
Fxx = [diff(Fx, x(1)); diff(Fx, x(2)); diff(Fx, x(3))]
```

Fxx =

$$\begin{pmatrix} -\sigma_2 - 2\,x_3^2\,z\,\sigma_3 & -\sigma_2 - 2\,x_3\,z\,\sigma_3 & \sigma_1 \\ -\sigma_2 - 2\,x_3\,z\,\sigma_3 & -2\,z\,\sigma_3 - \sigma_2 & -\sigma_2 - 2\,x_1\,z\,\sigma_3 \\ \sigma_1 & -\sigma_2 - 2\,x_1\,z\,\sigma_3 & -\sigma_2 - 2\,x_1^2\,z\,\sigma_3 \end{pmatrix}$$

where

$$\sigma_1 = 2\,z\cos(x_2 + x_1\,x_3) - \sigma_2 - 2\,x_1\,x_3\,z\,\sigma_3$$

$$\sigma_2 = \cos(x_1 + x_2 + x_3)\,\sin(z)$$

$$\sigma_3 = \sin(x_2 + x_1\,x_3)$$

```
Fxz = diff(Fx, z).'
```

Fxz =

$$\begin{pmatrix} 2\,x_3\cos(x_2 + x_1\,x_3) - \sin(x_1 + x_2 + x_3)\,\cos(z) \\ 2\cos(x_2 + x_1\,x_3) - \sin(x_1 + x_2 + x_3)\,\cos(z) \\ 2\,x_1\cos(x_2 + x_1\,x_3) - \sin(x_1 + x_2 + x_3)\,\cos(z) \end{pmatrix}$$

```
Fzx = [diff(Fz, x(1)), diff(Fz, x(2)), diff(Fz, x(3))]
```

Fzx = $\big(2\,x_3\cos(x_2 + x_1\,x_3) - \sin(x_1 + x_2 + x_3)\,\cos(z) \quad 2\cos(x_2 + x_1\,x_3) - \sin(x_1 + x_2 + x_3)\,\cos(z) \quad 2\,x_1\cos(x_2 + x$

```
Fzz = diff(Fz, z)
```

Fzz = $-\cos(x_1 + x_2 + x_3)\,\sin(z)$

```
d2zdx2 = Fz^-2 * Fx.' * (Fzx + Fzz*dzdx) - Fz^-1 * (Fxx + Fxz*dzdx);
pxx = [diff(px, x(1)); diff(px, x(2)); diff(px, x(3))];
pxz = diff(px, z).';
pzx = [diff(pz, x(1)), diff(pz, x(2)), diff(pz, x(3))];
pzz = diff(pz, z);
d2pdy2 = pxx + (pxz*dzdx + dzdx.'*pzx) + pzz*(dzdx.')*dzdx + pz*d2zdx2
```

d2pdy2 =

$$\left(\begin{array}{c}\frac{4\,x_2\,x_3\,z\cos(\sigma_{25})\,\sigma_{24}}{\sigma_{23}}-\frac{{\sigma_{24}}^2\,\sigma_{20}}{{\sigma_{23}}^2}-2\,z\sin(\sigma_{25})\left(\frac{\sigma_{27}\sin(z)+2\,{x_3}^2\,z\,\sigma_{26}-\frac{\sigma_4\,\sigma_{24}}{\sigma_{23}}}{\sigma_{23}}+\frac{\sigma_{24}\,\sigma_1}{{\sigma_{23}}^2}\right)-{x_2}^2\,{x_3}^2\cos(\sigma_{25})\\ x_3\sin(\sigma_{25})-2\,z\sin(\sigma_{25})\left(\frac{\sigma_{27}\sin(z)-\frac{\sigma_{18}\,\sigma_{24}}{\sigma_{23}}+2\,x_3\,z\,\sigma_{26}}{\sigma_{23}}-\frac{\sigma_{22}\,\sigma_1}{{\sigma_{23}}^2}\right)+\sigma_7-\sigma_{17}+\sigma_{13}-\sigma_9\\ x_2\sin(\sigma_{25})+2\,z\sin(\sigma_{25})\left(\frac{2\,z\,\sigma_{28}-\sigma_{27}\sin(z)+\frac{\sigma_5\,\sigma_{24}}{\sigma_{23}}-\sigma_{19}}{\sigma_{23}}-\frac{\sigma_{21}\,\sigma_1}{{\sigma_{23}}^2}\right)-\sigma_{16}-\sigma_8+\sigma_{14}+\sigma_{10}\end{array}\right.$$

where

$$\sigma_1=\sigma_{29}\cos(z)-2\,x_3\,\sigma_{28}+\frac{\sigma_{27}\sin(z)\,\sigma_{24}}{\sigma_{23}}$$

$$\sigma_2=\sigma_{29}\cos(z)-2\,x_1\,\sigma_{28}+\frac{\sigma_{27}\sin(z)\,\sigma_{21}}{\sigma_{23}}$$

$$\sigma_3=2\,\sigma_{28}-\sigma_{29}\cos(z)+\frac{\sigma_{27}\sin(z)\,\sigma_{22}}{\sigma_{23}}$$

$$\sigma_4=2\,x_3\,\sigma_{28}-\sigma_{29}\cos(z)$$

$$\sigma_5=2\,x_1\,\sigma_{28}-\sigma_{29}\cos(z)$$

$$\sigma_6=\frac{\sigma_{22}\,\sigma_{21}\,\sigma_{20}}{{\sigma_{23}}^2}$$

$$\sigma_7=\frac{\sigma_{22}\,\sigma_{24}\,\sigma_{20}}{{\sigma_{23}}^2}$$

$$\sigma_8=\frac{\sigma_{21}\,\sigma_{24}\,\sigma_{20}}{{\sigma_{23}}^2}$$

$$\sigma_9=\frac{2\,x_2\,x_3\,z\cos(\sigma_{25})\,\sigma_{22}}{\sigma_{23}}$$

$$\sigma_{10}=\frac{2\,x_2\,x_3\,z\cos(\sigma_{25})\,\sigma_{21}}{\sigma_{23}}$$

$$\sigma_{11}=\frac{2\,x_1\,x_3\,z\cos(\sigma_{25})\,\sigma_{21}}{\sigma_{23}}$$



$$\sigma_{12}=\frac{2\,x_1\,x_2\,z\cos(\sigma_{25})\,\sigma_{22}}{\sigma_{23}}$$

```
func2 = matlabFunction(d2pdy2)
```

```
func2 = function_handle with value:
    @(x_1,x_2,x_3,z)reshape([-x_2.^2.*x_3.^2.*cos(z.^2-x_1.*x_2.*x_3)-1.0./(sin(x_2+x_1.*x_3).*2.0+cos(x_1
```

```
z_test = 1.543295599106779
```

```
z_test = 1.5433
```

```
func2(1, 2, 3, z_test)
```

```
ans = 3×3
  -22.2226   -4.9360   -7.1680
   -4.9360   -0.9918   -1.5241
   -7.1680   -1.5241   -2.0855
```

```
% tic; func(1,2,3,z_test); toc
cplxDiff(@(x) func(x(1), x(2), x(3), z_test), [1;2;3], 1e-20)
```

```
ans = 3×3
  -14.4862   -1.8615   -4.0935
    0.6120    1.2130    0.6807
   -5.8336   -0.9938   -1.5551
```

```
% aaa(funcF, funcp, [1,2,3], z_test)
norm(func2(1, 2, 3, z_test), 'fro')
```

```
ans = 25.5990
```

```
norm(cplxDiff(@(x) func(x(1), x(2), x(3), z_test), [1;2;3], 1e-20), 'fro')
```

```
ans = 16.4261
```

```
% tic; aaa(funcF, funcp, [1,2,3], z_test); toc
```

```
function fp =  cplxDiff(f, x0, h)
    L = length(x0);
    sz_func = size(f(x0));
    sz_inpt = size(x0);
    for i = 1:L
        H = zeros(size(x0));
        H(i) = h*1i;
        fp(:, i) = imag(f(x0 + H))/h;
    end
end

function out = aaa(F, p, x0, z0)
    h = 1e-20;
    Fx = cplxDiff(@(x) F(x, z0), x0, h);
    Fz = cplxDiff(@(z) F(x0, z), z0, h);
    px = cplxDiff(@(x) p(x, z0), x0, h);
    pz = cplxDiff(@(z) p(x0, z), z0, h);

    out = px - pz*(Fx/Fz);
```

```
end
```