Problem 1: Computing

Problem Statement Using the following equation complete parts 1-5:

$$P = \cos\left(z^2 - x_1 x_2 x_3\right)$$

Part 1.1

Problem Statement Using a symbolic manipulator, automate the computation of the first derivative of P with respect to **x**, $\frac{\partial P}{\partial \mathbf{x}}$ or $P_{\mathbf{x}}$. Assuming $z = 4$. Show the resulting $P_{\mathbf{x}}$ for $\mathbf{x} = [1, 2, 3]^T$.

Take the partial derivative of P with respect to each index of $\vec{x}$ results in the following vector $P_{\mathbf{x}}$:

$$P_{\mathbf{x}} = \begin{bmatrix} x_2 x_3 \sin\left(z^2 - x_1 x_2 x_3\right) \\ x_1 x_3 \sin\left(z^2 - x_1 x_2 x_3\right) \\ x_1 x_2 \sin\left(z^2 - x_1 x_2 x_3\right) \end{bmatrix}$$

Plugging in the values for $\mathbf{x} = [1, 2, 3]^T$, results in the following vector: $P_{\mathbf{x}} = [-3.26413, -1.63206, -1.08804]^T$

Part 1.2

Problem Statement Use complex step differentiation to achieve the same gradient as in Part 1. Show the result and absolute error with respect to previous results.

$$f' = \frac{\text{imag}\left(f\left(x_0 + hi\right)\right)}{h}$$

Using the above equation to calculate the partial derivative, $P_{\mathbf{x}}$, a step of h in the imaginary direction is taken for each x_i and then evaluated above. Using an $h = 10^{-20}$, resulted in the follow vector $P_{\mathbf{x}} = [-3.26413, -1.63206, -1.08804]^T$ and an absolute error of 0 (I believe this is due to how Julia shortens values). Not until an $h = 10^{-11}$ does the error term become on the order of 10^{-16}.

Part 1.3

Problem Statement Use a 3, 5, and 7 point central difference stencil to approximate the same gradient using finite differentiating, at a step size of $h = 10^{-4}$. Show formulas used, the approximated gradients, and their absolute errors with respect to Part 1.

$$f'_{\text{3pt}} = \frac{-f(x_0-h)+f(x_0+h)}{2h}$$
$$f'_{\text{5pt}} = \frac{f(x_0-2h)-8f(x_0-h)+f(x_0+h)-f(x_0+2h)}{12h}$$
$$f'_{\text{7pt}} = \frac{-f(x_0-3h)+9f(x_0-2h)-45f(x_0-h)+45f(x_0+h)-9f(x_0+2h)+f(x_0+3h)}{60h}$$

Using the above equations for each of the stencils, the following gradients and errors were calculated.

$$f'_{\text{3pt}} = [-3.2641264694904, -1.6320633081884, -1.0880422145226]^T\,; \quad \varepsilon_{\text{3pt}} = 1.9750 \times 10^{-7}$$
$$f'_{\text{5pt}} = [-3.2641266653377, -1.6320633326697, -1.0880422217761]^T\,; \quad \varepsilon_{\text{5pt}} = 3.3971 \times 10^{-12}$$
$$f'_{\text{7pt}} = [-3.264126665337, -1.6320633326698, -1.0880422217775]^T\,; \quad \varepsilon_{\text{7pt}} = 2.6353 \times 10^{-12}$$

Part 1.4

Problem Statement Use a symbolic manipulator to automate the computation of the second derivative of P with respect to **x**, assuming z = 4. Show the resulting $P_{x_1x_1}$ or $\frac{\partial^2 P}{\partial \mathbf{x}^2}(1,1)$, evaluated at $\mathbf{x} = [1,2,3]^T$

Symbolically taking the partial derivative of P_{x_1} results in the following equation for $P_{x_1x_1}$:

$$P_{x_1x_1} = -{x_2}^2{x_3}^2\cos\left(z^2 - x_1x_2x_3\right)$$

Evaluating the equation for $\mathbf{x} = [1,2,3]^T$ yields this analytical derivative value $P_{x_1x_1} = 30.206575046752288$.

Part 1.5

Problem Statement Use the complex step method to differentiate the results of Part 1 in order to approximate the Hessian results in Part 4. Show the result and the absolute error with respect to Part 4.

By substituting P_{x_1} into the complex derivative formula for f gives us a numerical solution: $P_{x_1x_1} = 30.206575046752285$ with an error of $3.552713678800501e - 15$ for an $h = 10^{-20}$.

Problem 2

Problem Statement Consider the following initial conditions for a two-body orbit. μ is the standard gravitational parameter. Units are generic length and time units (LU and TU).

$$\mu = 1\,\frac{LU^3}{TU^2}; \quad \mathbf{r}_0 = \begin{bmatrix} 1 \\ 1 \\ 0.1 \end{bmatrix} LU; \quad \mathbf{v}_0 = \begin{bmatrix} 0.1 \\ 0.7 \\ 0.3 \end{bmatrix} \frac{LU}{TU}; \quad \mathbf{x}_0 = \begin{bmatrix} \mathbf{r}_0 \\ \mathbf{v}_0 \end{bmatrix}$$

Part 2.1

Problem Statement What is the period of the orbit in TU.

$$a = \frac{\mu\,\|\vec{r}\|}{2\,\mu - \|\vec{r}\|\,\|\vec{v}\|^2}$$

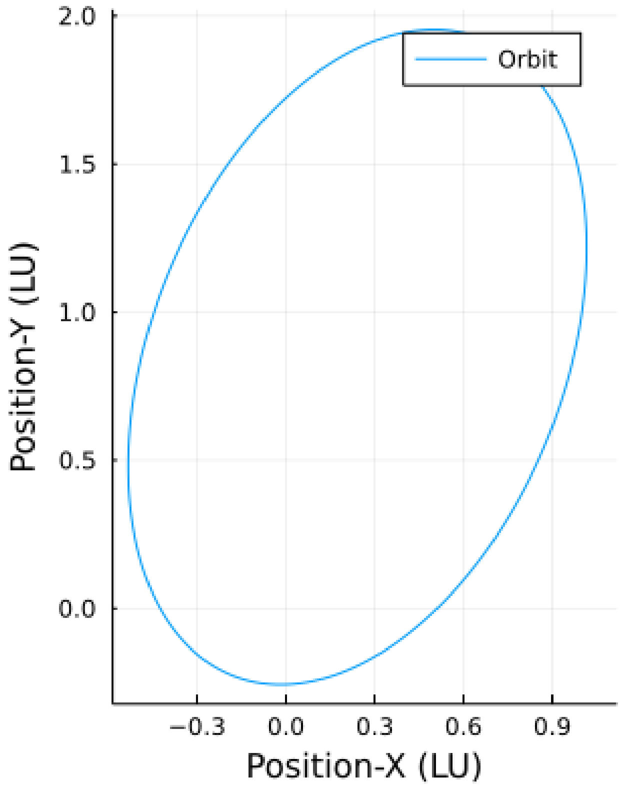
$$T = 2\pi\sqrt{\frac{a^3}{\mu}}$$

Using the above equations, the period of the above orbit was found to be $T = 8.4510(...)$ TU

Part 2.2

Problem Statement Use a variable step ODE solver with tight tolerances to compute and plot 1 period of the orbit in the xy-plane. Verify periodicity by reporting the norm of the vector difference between the beginning and end states. Change ODE tolerances to reduce this error to under 10^{-13}. Report the order/type of the integrator used and the tolerances needed.

State propagated using Tsit5 variable time solver (from Julia DifferentialEquations package), in order to achieve required error, absolute and relative tolerances of 10^{-13} were used.



Part 2.3

Problem Statement Using the final tolerance from above, compute the first order state transition matrix for the computed 1 period orbit using:

1) Complex Step Method
2) Variational Method
3) Finite Difference with 3 Point Stencil
4) Finite Difference with 5 Point Stencil

Part 2.4

Problem Statement Report the full 6x6 matrix for Part 2.3.1.

$$[\phi_a(T, t_0)] \approx \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Part 2.5

> **Problem Statement** Report the Frobenius norm of the matrix in Part 2.4.

$$\| [\phi_a(T, t_0)] \| = 2.4494897(...)$$

Part 2.6

> **Problem Statement** Compute the Frobenius norm of the matrix difference of the Parts 2.3.2-2.3.4 with respect to Part 2.3.1.

$$\Delta [\phi_b(T, t_0)] = 25.872959(...)$$

Note on this number: I understand this is obviously wrong, they all should be (relatively) equal. I've tried this method in both MATLAB, Julia, and confirmed my equations with papers, so I am not exactly sure what I did wrong. Most likely a simple programmatic error rather than something with my math. I'll explain it in the Appendix.

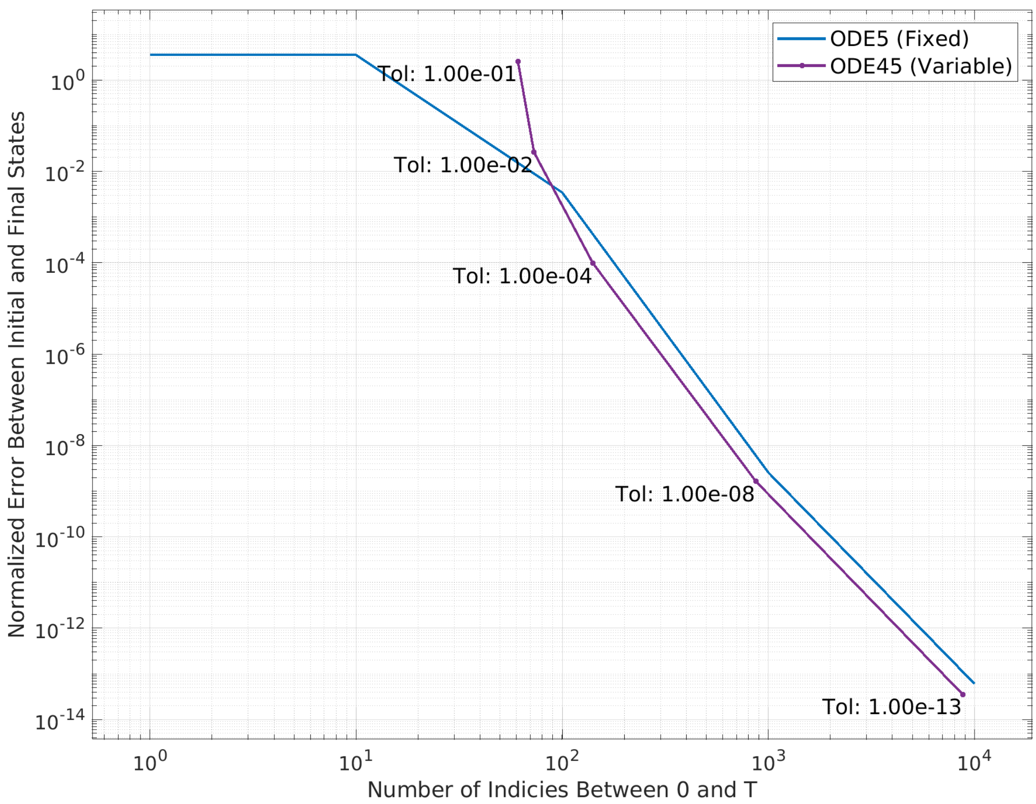
$$\Delta [\phi_c(T, t_0)] = 0.002676(...)$$

$$\Delta [\phi_d(T, t_0)] = 0.003485(...)$$

Part 2.7

> **Problem Statement** Experiment with fixed vs variable step integrators, at different tolerances. Report on the accuracy of the STMs by repeating Part 2.6, treating the complex STM as truth. Discuss the results.

Comparing the total number of steps taken, regardless of size, any solution of reasonable accuracy, error under 10^{-4}, shows that the variable step solves take less total steps to solve the solution than their fixed time step counterparts. From my understanding of how these solvers work, the variable step solvers take larger steps when their are more sure of future gradients estimates. Thus “stepping over” steps required to be taken by the fixed time step solvers.



Part 2.8

Problem Statement Compute the full Hessian, $\frac{\partial^2 KE}{\partial \mathbf{x}_0^2}$, of the kinetic energy KE at $t = t_0 + 5\,TU$ with respect to the initial state using:

1) Variational Method plus Chain Rule
2) Finite Difference of Your Choice (Show Formula)

$$f''_{\text{5pt}} = \frac{-f(x_0 - 2h) + 16f(x_0 - h) - 30f(x_0) + 16f(x_0 + h) - f(x_0 + 2h)}{12h^2}$$

Part 2.9

Problem Statement Report the full 6×6 matrix for Part 2.8.1.

$$\frac{\partial^2 KE}{\partial \mathbf{x}} = \begin{bmatrix} 30.8753 & 39.755 & 10.5566 & 17.4601 & 86.5413 & 25.1257 \\ 39.755 & 45.9582 & 15.0279 & 24.6 & 97.8826 & 32.1899 \\ 10.5566 & 15.0279 & -2.0744 & 2.36418 & 49.7998 & 5.08131 \\ 17.4601 & 24.6 & 2.36418 & 3.07213 & 68.5336 & 14.036 \\ 86.5413 & 97.8826 & 49.7998 & 68.5336 & 170.047 & 80.2224 \\ 25.1257 & 32.1899 & 5.08131 & 14.036 & 80.2224 & 14.8214 \end{bmatrix}$$

Due to the issues in Part 2.6, the values shown will be from the Finite Difference Method.

Part 2.10

Problem Statement Report the Frobenius norm of the matrix from Part 2.9.

$$\|\frac{\partial^2 KE}{\partial \mathbf{x}}\| = 320.97984(...)$$

Due to the issues in Part 2.6, the values shown will be from the Finite Difference Method.

Appendix

Variational Equations STM Method

Following both the method in Bates, and your lecture notes I was able to match the equations for the A matrix. (Brackets around variables indicate matrices)

$$[\mathsf{A}] = \begin{bmatrix} [0] & [\mathbf{I}] \\ [\mathsf{G}] & [0] \end{bmatrix}_{6x6}$$

$$[\mathsf{G}] = \frac{\mu}{\|\vec{r}\|^5}\left(3\,\vec{r}\,\vec{r}^T - \|\vec{r}\|^2\,[\mathbf{I}]\right)$$

$$\left[\dot{\phi}\right] = [\mathsf{A}]\,[\phi] \qquad [\phi]_0 = [\mathbf{I}]_{6x6}$$

MATLAB Code Tested .

```
opts = odeset('reltol', 1e-13, 'abstol', 1e-13);
[t, Y] = ode45(f, [0 T], [x; reshape(eye(6), [], 1)], opts);

function du = EoM(u, t, mu)
    rvec = u(1:3);
    r = norm(rvec);
    rhat = rvec/r;
    vvec = u(4:6);
    phi = reshape(u(7:42), 6, 6);

    du(1:3) = vvec;
    du(4:6) = -mu*(rvec/r^3);

    G = mu/r^5 * ( (3 * (rvec*rvec.')) - (r^2*eye(3)) );

    zmtrx = zeros(3, 3);

    A = [zmtrx eye(3); G zmtrx];
    du(7:42) = reshape(A*phi, [], 1);
    du = du(:);
end
```

Output

$$\phi_b = \begin{bmatrix} -0.0841 & -1.0841 & -0.1084 & -0.3089 & -2.1625 & -0.9268 \\ -7.5885 & -6.5885 & -0.7588 & -2.1625 & -15.1373 & -6.4874 \\ -3.2522 & -3.2522 & 0.6748 & -0.9268 & -6.4874 & -2.7803 \\ 3.8042 & 3.8042 & 0.3804 & 2.0841 & 7.5885 & 3.2522 \\ 3.8042 & 3.8042 & 0.3804 & 1.0841 & 8.5885 & 3.2522 \\ 0.3804 & 0.3804 & 0.0380 & 0.1084 & 0.7588 & 1.3252 \end{bmatrix}$$

I'm not sure why this isn't symmetric, I've done variational equations for the three body problem following this exact method, and did not encounter this problem. Although on a side note, the eigenvalues of this matrix (in diagonal form from `eig`) did return an 6x6 identity matrix. So I believe it is mostly correct, I am just missing one final piece.

Julia Notebook Code

(Next Page)

Problem 1

Part 1

```
begin
    # PUBLIC PACKAGES
    using Symbolics ✓, LinearAlgebra ✓, DifferentialEquations ✓

    # PRIVATE PACKAGES
    include("/home/burtonyale/Documents/repos
/OptimalSpacecraftTrajectories/src/OptimalSpacecraftTrajectories.jl")
    import .OptimalSpacecraftTrajectories
    const OST = OptimalSpacecraftTrajectories

    md"""### Problem 1
    #### Part 1"""
end
```

```
using Plots ✓
```

$$\cos\left(16 - x_1 x_2 x_3\right)$$

```
begin
    @variables x[1:3]
    z = 4
    f = cos(-x[1]*x[2]*x[3] + z^2)
end
```

f′ = Symbolics.Num[

1: $\sin\left(16 - x_1 x_2 x_3\right) x_2 x_3$

2: $\sin\left(16 - x_1 x_2 x_3\right) x_1 x_3$

3: $\sin\left(16 - x_1 x_2 x_3\right) x_1 x_2$

]

```
f′ = Symbolics.derivative.(f, [x[1], x[2], x[3]]; simplify=true)
```

```julia
f′il = build_function(f′, x, expression=false)[1];
```

```
f′_actual = Float64[
                1:  -3.26413
                2:  -1.63206
                3:  -1.08804
            ]
```

```julia
f′_actual = f′il([1, 2, 3])
```

Part 2

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

```julia
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
```

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

```julia
func(x) = cos(-x[1]*x[2]*x[3] + z^2)
```

```
0.0
```

```julia
norm(f′_actual - cplxDiff(func, [1.0, 2.0, 3.0], 1e-20))
```

Part 3

cntrDiff (generic function with 1 method)

```
function cntrDiff(f, x₀, h, N)
    f′ = []
    L = length(x₀)

    if N == 3
        g = (x, δx) -> (f(x + δx) - f(x - δx))/(2*h)
    elseif N == 5
        g = (x, δx) -> (f(x - 2*δx) - 8*f(x - δx) + 8*f(x + δx) - f(
2*δx))/(12*h)
    elseif N == 7
        g = (x, δx) -> (-f(x - 3*δx) + 9*f(x - 2*δx) - 45*f(x - δx)
45*f(x + δx) - 9*f(x + 2*δx) + f(x + 3*δx))/(60*h)
    end

    for i = 1:L
        δx = zeros(L)
        δx[i] = h
        push!(f′, g(x₀, δx))
    end
    return f′
end
```

-3.2641264694904804 -1.6320633081884361 -1.0880422145226332

```
md"$(cntrDiff(func, [1.0, 2.0, 3.0], 1e-4, 3))"
```

-3.2641266653377072 -1.6320633326697789 -1.0880422217761827

```
md"$(cntrDiff(func, [1.0, 2.0, 3.0], 1e-4, 5))"
```

-3.264126665337763 -1.6320633326698715 -1.0880422217775336

```
md"$(cntrDiff(func, [1.0, 2.0, 3.0], 1e-4, 7))"
```

Part 4

f″ =

$$-{x_3}^2{x_2}^2\cos\left(16 - x_1x_2x_3\right)$$

```
f″ = Symbolics.derivative.(f′[1], x[1]; simplify=true)
```

```
f″jl = build_function(f″, x, expression=false);
```

f″_actual = 30.206575046752288

```
f″_actual = f″jl([1.0, 2.0, 3.0])
```

```
3.552713678800501e-15
```

```
f″_actual - cplxDiff(f′il, [1.0, 2.0, 3.0], 1e-20)[1][1]
```

Problem 2

Part 1

Period = 8.451040406976288 TU

```
begin
    μ = 1 # LU³/TU²
    x₀ = [1.0, 1.0, 0.1, # LU
          0.1, 0.7, 0.3] # LU/TU
    a = 1 / (2/norm(x₀[1:3]) - (norm(x₀[4:6])^2)/μ)
    # angmom = norm(x₀[1:3]×x₀[4:6])
    # rad = x₀[1:3]; vel = x₀[4:6]
    # velr = rad⋅vel / norm(rad)
    # e = norm((1/μ) * ((norm(vel)^2 - μ/norm(rad)^2) .- norm(rad)*velr*vel))
    # a = angmom^2 / μ / (1-e^2)
    T = 2*π*sqrt(a^3 / μ)
    md"Period = $(T) TU"
    # md"$(a)"
end
```

Part 2

```
EoM! (generic function with 1 method)
```

```
function EoM!(du, u, p, t)
    μ, = p
    r⃗ = u[1:3]
    v⃗ = u[4:6]
    du[1:3] = v⃗
    du[4:6] = -μ*(r⃗/norm(r⃗)^3)
    # Φ = reshape(u[7:end], (6, 6))
    # A = μ*
    return du
end
```

```
prob =
ODEProblem with uType Vector{Float64} and tType Float64. In-place: true
timespan: (0.0, 8.451040406976288)
u0: 6-element Vector{Float64}:
 1.0
 1.0
 0.1
 0.1
 0.7
 0.3
```

```julia
prob = ODEProblem(EoM!, x₀, (0.0, T), (μ))
```

Integrator: Tsit5 | Tol: 1e-13 | Miss: 1.5771215576378597e-13 LU

```julia
sol = solve(prob, reltol=1e-13, abstol=1e-13); md"Integrator: Tsit5 | Tol: 1e-13 | Miss: $(norm(sol[1][1:3] - sol[end][1:3])) LU"
```

```julia
plot(sol, vars=(1,2),
    xlabel="Position-X (LU)", ylabel="Position-Y (LU)", zlabel="Position-Z (LU)",
label="Orbit", format=:png, aspect_ratio=:equal); savefig("~/Documen
/repos/OptimalSpacecraftTrajectories/src/HW/hw1_2.2eq.png")
```

Part 3

Function Setup

```julia
begin
    function genSTM(diffFunc, EoM, x₀, xₜ, h)
        L = length(x₀)
        ∂x₀ = reshape(hcat(diffFunc(EoM, x₀, h)...), (L, L))
        ∂xₜ = reshape(hcat(diffFunc(EoM, xₜ, h)...), (L, L))
        Φₜ = ∂xₜ/∂x₀
    end
    EoM_shrthnd1 = (u) -> EoM!(zeros(Complex{Float64}, 6), u, (μ), (
T))
    EoM_shrthnd2 = (u) -> EoM!(zeros(6), u, (μ), (0.0, T))
    EoM_shrthnd1 = (u) -> EoM!(zeros(Complex{Float64}, 6), u, (μ), (
T))
    EoM_shrthnd2 = (u) -> EoM!(zeros(6), u, (μ), (0.0, T))
    EoM_shrthnd3 = (u) -> EoM!(zeros(6), u, (μ), (0.0, T))
    md"Function Setup"
end
```

Calculating STMs

```
begin
    # COMPLEX STEP METHOD
    Φa = genSTM(cplxDiff, EoM_shrthnd1, x₀, sol[end], 1e-13);

    # VARIATIONAL METHOD
    ⊗(u, v) = u*transpose(v)
    function EoMSTM!(du, u, p, t)
        μ, = p
        r⃗ = u[1:3]
        r = sqrt(u[1]^2 + u[2]^2 + u[3]^2)
        v⃗ = u[4:6]
        Φ = reshape(u[7:42], (6, 6))

        Idnt = [1.0 0 0; 0 1 0; 0 0 1]
        G = μ/r^5 * ( (3 * r⃗⊗r⃗) - (r^2 * Idnt) ) # ← from Bates
        # G = OST.jacobian(r⃗, μ) # Same value as above
        zmtrx = 0.0*Idnt

        A = [zmtrx Idnt; G zmtrx]
        Φ̇ = A*Φ

        du[1:3] = v⃗
        du[4:6] = -μ*(r⃗/r^3)
        du[7:42] = reshape(Φ̇, 36)
        return du
    end
    probb = ODEProblem(EoMSTM!, vcat(x₀, reshape(1.0*Matrix(I, 6, 6)
36)), (0.0, T), (μ))
    solb = solve(probb, reltol=1e-8, abstol=1e-8);
    Φb = reshape(solb[end][7:42], (6, 6));

    # 3 POINT FINITE DIFFERENCE
    Φc = genSTM((f, x, h) -> cntrDiff(f, x, h, 3), EoM_shrthnd2, x₀,
sol[end], 1e-13)

    # 5 POINT FINITE DIFFERENCE
    Φd = genSTM((f, x, h) -> cntrDiff(f, x, h, 5), EoM_shrthnd2, x₀,
sol[end], 1e-13)
    md"Calculating STMs"
end
```

Part 4

```
6×6 Matrix{Float64}:
  1.0  0.0  0.0  -0.0  -0.0  -0.0
  0.0  1.0  0.0  -0.0  -0.0  -0.0
  0.0  0.0  1.0  -0.0  -0.0  -0.0
  0.0  0.0  0.0   1.0  -0.0   0.0
 -0.0  0.0  0.0  -0.0   1.0   0.0
 -0.0  0.0  0.0  -0.0  -0.0   1.0
```

```julia
begin
    ∂x₀ = reshape(hcat(cplxDiff((u) -> EoM!(zeros(Complex{Float64}, u, (μ), (0.0, T)), x₀, 1e-13)...), (6, 6))
    ∂xₜ = reshape(hcat(cplxDiff((u) -> EoM!(zeros(Complex{Float64}, u, (μ), (0.0, T)), sol[end], 1e-13)...), (6, 6))
    Φ = ∂xₜ/∂x₀
end
```

Part 5

```
2.4494897427834053
```

```julia
norm(Φ)
```

Part 6

{a,b} = 25.87295976748877 ← I've double checked my method in MATLAB and usin couple of papers, I am unsure where this error is coming from

{a,c} = 0.0026762560780534987

{a,d} = 0.003485831114903648

```julia
md"""#### Part 6
{a,b} = $(norm(Φb - Φa)) ← I've double checked my method in MATLAB a using a couple of papers, I am unsure where this error is coming fro

{a,c} = $(norm(Φc - Φa))

{a,d} = $(norm(Φd - Φa))"""
```

Part 7

```
ode5 (generic function with 1 method)
```

```julia
function ode5(f, tspan, y₀)
    h = diff(tspan)[1];
    neq = length(y₀)
    N = length(tspan)
    Y = zeros(neq, N)

    C = [0.2, 0.3, 0.8, 8/9, 1]
    A = transpose([0.2 0 0 0 0;
         3/40 9/40 0 0 0;
         44/45 -56/15 32/9 0 0;
         19372/6561 -25360/2187 64448/6561 -212/729 0;
         9017/3168 -355/33 46732/5247 49/176 -5103/18656])
    B = [35/384, 0, 500/1113, 125/192, -2187/6784, 11/84]

    nstages = length(B)
    F = zeros(neq, nstages)

    Y[:, 1] = y₀
    for i = 2:N
        ti = tspan[i-1]
        hi = h[i-1]
        yi = Y[:, i-1]

        F[:, 1] = f(ti, yi)
        for stageccc = 2:nstages
            tstage = ti + C[stage-1]*hi
            ystage = ti + F[:, 1:stage-1]*(hi*A[1:stage-1, stage-1])
            F[:, stage] = f(tstage, ystage)
        end
        Y[:, i] = yi + F*(hi*B)
    end
    Y = transpose(Y)
end
```

```julia
# On MATLAB
```

Part 8

$$f^{\prime\prime} = -\frac{f_{-2} - 16f_{-1} + 30f_{0} - 16f_{1} + f_{2}}{-12h^2}$$

```
function cntrDiff′(f, x₀, h)
    f″ = []
    L = length(x₀)

    for i = 1:L, j = 1:L
        δx = δy = zeros(L)
        δx[i] += h
        δy[j] += h
        # push!(f″, ( f(x₀ + δx + δy) - f(x₀ + δx) - f(x₀ + δy) + f(
)/h^2)
        push!(f″, -(f(x₀ - 2*δx) - 16*f(x₀ - δx) + 30*f(x₀) - 16*f(x
δx) + f(x₀ + 2*δx))/(12*h^2))
    end
    f″ = reshape(f″, (6, 6))
    return f″
end; md"$f^{\prime\prime}  = -\frac{f_{-2} - 16f_{-1} + 30f_{0} - 16
+ f_{2}}{-12h^2}$"
```

Part 9

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

```
function KE(u, μ, T, idx)
    sol = solve(ODEProblem(EoM!, u, (0.0, T), (μ)), reltol=1e-13,
abstol=1e-13)
    if idx == -1
        r⃗ = sol[end][1:3]
        v⃗ = sol[end][4:6]
    else
        r⃗ = sol[idx][1:3]
        v⃗ = sol[idx][4:6]
    end
    εₖ = 0.5*norm(v⃗)^2
end
```

```
0.29499999999996035
```

```
KE(x₀, μ, T, -1)
```

```
6×6 Matrix{Any}:
 30.8753  39.755   10.5566   17.4601    86.5413  25.1257
 39.755   45.9582  15.0279   24.6       97.8826  32.1899
 10.5566  15.0279  -2.0744    2.36418   49.7998   5.08131
 17.4601  24.6      2.36418   3.07213   68.5336  14.036
 86.5413  97.8826  49.7998   68.5336   170.047   80.2224
 25.1257  32.1899   5.08131  14.036     80.2224  14.8214
```

```
begin
    # ∂²x₀ = cntrDiff′((u) -> KE(u, μ, T, 1), x₀, 1e-6)
    ∂²xₜ = cntrDiff′((u) -> KE(u, μ, 5, -1), x₀, 1e-6)
    # hes = ∂²xₜ/∂²x₀
end
```

Part 10

```
320.97984910875914
```

```
norm(∂²xₜ)
```