Exercise with matrix exponential

```
> restart;
> with(LinearAlgebra):
```

A 3x3 matrix as example

```
> A    := <<1,1,0>|<0,1,-1>|<1,1,-1>>;
  Imat := IdentityMatrix(3) ;
```

$$A := \begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 1 \\ 0 & -1 & -1 \end{bmatrix}$$

$$Imat := \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Characteristic polynomial

```
> pchar := Determinant(A-lambda*Imat);
```

$$pchar := \lambda^2 - \lambda^3 - 1$$

Compute eigenvalues

```
> RES := evalf(solve([pchar],[lambda]));
```

$$RES := [[\lambda = -0.7548776667], [\lambda = 0.8774388331 - 0.7448617670\,I], [\lambda = 0.8774388331 + 0.7448617670\,I]]$$

Compute interpolating polinomial

```
> EQ1 := subs(RES[1],a0+a1*lambda+a2*lambda^2-exp(lambda));
  EQ2 := subs(RES[2],a0+a1*lambda+a2*lambda^2-exp(lambda));
  EQ3 := subs(RES[3],a0+a1*lambda+a2*lambda^2-exp(lambda));
```

$$EQ1 := a0 - 0.7548776667\,a1 + 0.5698402917\,a2 - e^{-0.7548776667}$$

$$EQ2 := a0 + (0.8774388331 - 0.7448617670\,I)\,a1 + (0.2150798539 - 1.307141279\,I)\,a2 - e^{0.8774388331 - 0.7448617670\,I}$$

$$EQ3 := a0 + (0.8774388331 + 0.7448617670\,I)\,a1 + (0.2150798539 + 1.307141279\,I)\,a2 - e^{0.8774388331 + 0.7448617670\,I}$$

```
> RESP := solve( {EQ||(1..3)}, {a||(0..2)} ) ;
```

$$RESP := \{a2 = 0.7064956200, a1 = 0.9486419977, a0 = 0.7835871038\}$$

Compute martrix exponential

```
> expA := subs(RESP,a0*Imat+a1*A+a2*A.A) ;
```

$$expA := \begin{bmatrix} 2.438724722 & -0.7064956200 & 0.9486419977 \\ 2.361633238 & 1.732229102 & 1.655137618 \\ -0.7064956200 & -0.9486419977 & -0.1650548939 \end{bmatrix}$$

Exponential computed with MATLAB

```
> expA1 := <<2.438724721694499,2.361633237142482,
  -0.706495619699671>|
```

```
        <-0.706495619699672,1.732229101994827,
-0.948641997743139>|
        <0.948641997743139,1.655137617442810,
-0.165054893491450>>;
```

$$expA1 := \begin{bmatrix} 2.438724721694499 & -0.706495619699672 & 0.948641997743139 \\ 2.361633237142482 & 1.732229101994827 & 1.655137617442810 \\ -0.706495619699671 & -0.948641997743139 & -0.165054893491450 \end{bmatrix}$$

Check computation

```
> expA-expA1 ;
```

$$\begin{bmatrix} 3.05500957864524026\ 10^{-10} & -3.00327984703585571\ 10^{-10} & -4.31389368671375450\ 10^{-11} \\ 8.57518056562867059\ 10^{-10} & 5.17297316093845439\ 10^{-12} & 5.57189849814676563\ 10^{-10} \\ -3.00328983904307734\ 10^{-10} & 4.31389368671375450\ 10^{-11} & -4.08549999386664808\ 10^{-10} \end{bmatrix}$$

Using Taylor serie

```
> N := 6 ;
```

$$N := 6$$

```
> app1 := evalf(Imat/N!):
> for k from N-1 by -1 to 0 do
    app1 := evalf(A.app1 + Imat / k!) ;
  end:
> app1 ;
```

$$\begin{bmatrix} 2.43888888899999978 & -0.706944444400000038 & 0.948611111199999967 \\ 2.36249999999999982 & 1.73194444500000011 & 1.65555555599999993 \\ -0.706944444499999935 & -0.948611111199999967 & -0.165277777999999986 \end{bmatrix}$$

Compute error of the approximation

```
> MatrixNorm(app1 - expA,infinity);
```

$$0.00156935699999971590$$

Division and step for ODE approximant of exponential matrix

```
> N  := 10 ;
  DT := 1/N ;
```

$$N := 10$$

$$DT := \frac{1}{10}$$

Using Euler scheme

Adavancing matrix for Euler

```
> B := Imat + DT*A ;
```

$$B := \begin{bmatrix} \frac{11}{10} & 0 & \frac{1}{10} \\ \frac{1}{10} & \frac{11}{10} & \frac{1}{10} \\ 0 & -\frac{1}{10} & \frac{9}{10} \end{bmatrix}$$

```
> X0 := Imat:
> for k from 1 to N do
    X||k := B.X||(k-1):
  end:
> app2 := evalf(X||N) ;
```

$$app2 := \begin{bmatrix} 2.423528480 & -0.5907650901 & 0.9762704702 \\ 2.157800650 & 1.832763390 & 1.567035560 \\ -0.5907650901 & -0.9762704702 & -0.1197775501 \end{bmatrix}$$

Compute error of the approximation

```
> MatrixNorm(app2-expA,infinity) ;
```

$$0.392468933999999825$$

Using Collatz scheme

```
> B := Imat + DT*A + (DT^2/2)*A.A;
```

$$B := \begin{bmatrix} \frac{221}{200} & -\frac{1}{200} & \frac{1}{10} \\ \frac{11}{100} & \frac{11}{10} & \frac{21}{200} \\ -\frac{1}{200} & -\frac{1}{10} & \frac{9}{10} \end{bmatrix}$$

```
> X0 := Imat :
> for k from 1 to N do
    X||k := B.X||(k-1):
  end:
> app3 := evalf(X||N) ;
```

$$app3 := \begin{bmatrix} 2.442167035 & -0.7036250958 & 0.9511038786 \\ 2.358354070 & 1.738541939 & 1.654728974 \\ -0.7036250958 & -0.9511038786 & -0.1636658179 \end{bmatrix}$$

Compute error of the approximation

```
> MatrixNorm(app3-expA,infinity) ;
```

$$0.0100006490000001946$$

Using Crank–Nicolson scheme

```
> B := evalf(LinearSolve(Imat - (DT/2)*A,Imat + (DT/2)*A)) ;
```

$$B := \begin{bmatrix} 1.104986186 & -0.005262465465 & 0.09998684384 \\ 0.1105117748 & 1.099723721 & 0.1052493093 \\ -0.005262465465 & -0.09998684384 & 0.8997500329 \end{bmatrix}$$

```
> X0 := Imat:
> for k from 1 to N do
    X||k := B.X||(k-1):
  end:
> app4 := X||N ;
```

$$app4 := \begin{bmatrix} 2.43675086378835148 & -0.707934760411127904 & 0.947260534015452515 \\ 2.36313005798259201 & 1.72881611348868302 & 1.65519529733151649 \\ -0.707934760458015731 & -0.947260536775599182 & -0.165704964039064484 \end{bmatrix}$$

Compute error of the approximation

```
> MatrixNorm(app4-expA,infinity) ;
```

$$0.00496748782542555389$$

The same but with squaring

```
> M  := 3 ;
  N  := 2^M ;
  DT := 1/N ;
```

$$M := 3$$

$$N := 8$$

$$DT := \frac{1}{8}$$

Using Euler scheme and squaring

```
> B := evalf(Imat + DT*A) ;
```

$$B := \begin{bmatrix} 1.125000000 & 0. & 0.1250000000 \\ 0.1250000000 & 1.125000000 & 0.1250000000 \\ 0. & -0.1250000000 & 0.8750000000 \end{bmatrix}$$

```
> app2 := B :
> for k from 1 to M do
    app2 := app2.app2:
  end:
```

Compute error of the approximation

```
> MatrixNorm(app2-expA,infinity) ;
```

$$0.483726588957122727$$

Using Collatz scheme and squaring

```
> B := evalf(Imat + DT*A + (DT^2/2)*A.A);
```

$$B := \begin{bmatrix} 1.132812500 & -0.007812500000 & 0.1250000000 \\ 0.1406250000 & 1.125000000 & 0.1328125000 \\ -0.007812500000 & -0.1250000000 & 0.8750000000 \end{bmatrix}$$

```
> app3 := B:
> for k from 1 to M do
    app3 := app3.app3:
  end:
```

Compute error of the approximation

```
> MatrixNorm(app3-expA,infinity) ;
                              0.0156251255288519797
```

Using Crank–Nicolson scheme and squaring

```
> B := evalf(LinearSolve(Imat - (DT/2)*A,Imat + (DT/2)*A)) ;
```

$$B := \begin{bmatrix} 1.132777922 & -0.008331163759 & 0.1249674564 \\ 0.1416297839 & 1.124446759 & 0.1332986202 \\ -0.008331163759 & -0.1249674564 & 0.8745118459 \end{bmatrix}$$

```
> app4 := B:
> for k from 1 to M do
    app4 := app4.app4:
  end:
```

Compute error of the approximation

```
> MatrixNorm(app4-expA,infinity) ;
                              0.00774752000050660250
```

Using PADE rational polinomial and squaring

```
> AT := DT*A:
> P := Imat+AT.(Imat/2+AT.(Imat/10+AT/120));
  Q := Imat+AT.(-Imat/2+AT.(Imat/10-AT/120)) ;
```

$$P := \begin{bmatrix} \frac{681}{640} & -\frac{97}{61440} & \frac{1}{16} \\ \frac{2017}{30720} & \frac{65279}{61440} & \frac{3937}{61440} \\ -\frac{97}{61440} & -\frac{1}{16} & \frac{57599}{61440} \end{bmatrix}$$

$$Q := \begin{bmatrix} \frac{601}{640} & -\frac{19}{12288} & -\frac{1}{16} \\ -\frac{365}{6144} & \frac{57601}{61440} & -\frac{749}{12288} \\ -\frac{19}{12288} & \frac{1}{16} & \frac{65281}{61440} \end{bmatrix}$$

```
> B := evalf(LinearSolve(Q,P)) ;
```

$$B := \begin{bmatrix} 1.132801808 & -0.008148187858 & 0.1249895679 \\ 0.1412859436 & 1.124653620 & 0.1331377557 \\ -0.008148187858 & -0.1249895679 & 0.8746744846 \end{bmatrix}$$

```
> app5 := B:
> for k from 1 to M do
    app5 := app5.app5:
  end:
```

Compute error of the approximation

```
> MatrixNorm(app4-expA,infinity) ;
                              0.00774752000050660250
```

Using Taylor serie and squaring

```
> NN := 3 ;
                                    NN := 3
> AT := A / N :
> app1 := evalf(Imat/NN!):
> for k from NN-1 by -1 to 0 do
    app1 := evalf(AT.app1 + Imat / k!) ;
  end:
> for k from 1 to M do
    app1 := app1.app1 ;
  end:
> app1 ;
```

$$\begin{bmatrix} 2.43902475089335313 & -0.706471793336023612 & 0.948826441245271378 \\ 2.36177002758614307 & 1.73255295486894445 & 1.65529823372565143 \\ -0.706471793549744764 & -0.948826440476450260 & -0.165099924551919991 \end{bmatrix}$$

Compute error of the approximation

```
> MatrixNorm(app1 - expA,infinity);
                              0.00062125818073899274 9
```