```
> ##################### P3 #################
makeJac:=proc(f)
local N,jac;
N:=LinearAlgebra:-Dimension(f);
jac:=VectorCalculus:-Jacobian(f,[seq(y[i],i=1..N)]);
unapply(jac,y);
end proc;
> Newton:=proc(f,y0,makeJac,tol,maxiter)
#global
local N,dy,Jac,F,yold,ynew,Err,iter;
N:=LinearAlgebra:-Dimension(f);
F:=unapply(f,y);
Jac:=makeJac(f);
yold:=y0;
dy:=LinearAlgebra:-LinearSolve(-Jac(yold),F(yold));
ynew:=yold+dy;
yold:=ynew;
Err:=10;
iter:=1;
while Err>tol and iter < maxiter do
iter:=iter+1;
dy:=LinearAlgebra:-LinearSolve(-Jac(yold),F(yold));
ynew:=yold+dy;
Err:=LinearAlgebra:-Norm(yold-ynew);
yold:=ynew;
end;
#[Err,ynew,iter];
ynew;
end proc;
```

$makeJac := \mathrm{proc}(f)$ ... end proc

local N, jac;

$N := LinearAlgebra$:-$Dimension(f)$;

$jac := VectorCalculus$:-$Jacobian(f, [\mathrm{seq}(y[i], i = 1 .. N)])$;

$\mathrm{unapply}(jac, y)$

$Newton := \mathrm{proc}(f, y0, makeJac, tol, maxiter)$

$\mathrm{local}\ N, dy, Jac, F, yold, ynew, Err, iter;$

$N := LinearAlgebra{:-}Dimension(f);$

$F := \mathrm{unapply}(f, y);$

$Jac := makeJac(f);$

$yold := y0;$

$dy := LinearAlgebra{:-}LinearSolve(-Jac(yold), F(yold));$

$ynew := yold + dy;$

$yold := ynew;$

$Err := 10;$

$iter := 1;$

$\mathrm{while}\ tol < Err\ \mathrm{and}\ iter < maxiter\ \mathrm{do}$

$iter := iter + 1;$

$dy := LinearAlgebra{:-}LinearSolve(-Jac(yold), F(yold));$

$ynew := yold + dy;$

$Err := LinearAlgebra{:-}Norm(yold - ynew);$

$yold := ynew$

$\mathrm{end\ do};$

$ynew$

$\mathrm{end\ proc}$

```
> eq1:=y[1]^2+y[2]^2+y[1]*y[2]-3;
eq2:=y[2]*y[2]+2*y[2]^2-2*(y[1]+y[2])-2;
eqns:=Vector(2,[eq1,eq2]);
y0:=Vector(2,[1.0,1.0]);
Newton(eqns,y0,makeJac,1e-6,50);
```

$$eq1 := y_1^2 + y_1 y_2 + y_2^2 - 3$$

$$eq2 := 3\,y_2^2 - 2\,y_1 - 2\,y_2 - 2$$

$$eqns := \begin{bmatrix} y_1^2 + y_1 y_2 + y_2^2 - 3 \\ 3\,y_2^2 - 2\,y_1 - 2\,y_2 - 2 \end{bmatrix}$$

$$y0 := \begin{bmatrix} 1.0 \\ 1.0 \end{bmatrix}$$

$$\begin{bmatrix} 0.537841544434349 \\ 1.39932526923185 \end{bmatrix}$$

```
> ######################### P4 Stability #####################
> restart;
> Digits:=15;
```

$$Digits := 15$$

```
> #dy/dt=lambda*y
eq4:=Y[1]=Y[0]+h*(O*lambda*Y[0]+(1-O)*lambda*Y[1]);
```

$$eq4 := Y_1 = Y_0 + h\,(O\,\lambda\,Y_0 + (1-O)\,\lambda\,Y_1)$$

```
> cons:=solve({eq4},{Y[1]});
```

$$cons := \{Y_1 = \frac{Y_0\,(O\,h\,\lambda+1)}{O\,h\,\lambda - h\,\lambda + 1}\}$$

```
> Y[0]:=1;
eqz:=Y[1]=subs(cons,Y[1]);
```

$$Y_0 := 1$$

$$eqz := Y_1 = \frac{O\,h\,\lambda+1}{O\,h\,\lambda - h\,\lambda + 1}$$

```
> eqz:=subs(lambda=z/h,Y[1]=theta,eqz);
```

$$eqz := \theta = \frac{O\,z+1}{O\,z - z + 1}$$

```
> z2:=solve(eqz,z);
```

$$z2 := -\frac{\theta-1}{O\,\theta - O - \theta}$$

```
> z2:=subs(theta=exp(I*phi),z2);
zz2:=unapply(z2,O);
zz2(1);
zz2(0);
zz2(1/2);
```

$$z2 := -\frac{e^{(\phi\,I)}-1}{O\,e^{(\phi\,I)} - O - e^{(\phi\,I)}}$$

$$zz2 := O \to -\frac{e^{(\phi\,I)}-1}{O\,e^{(\phi\,I)} - O - e^{(\phi\,I)}}$$

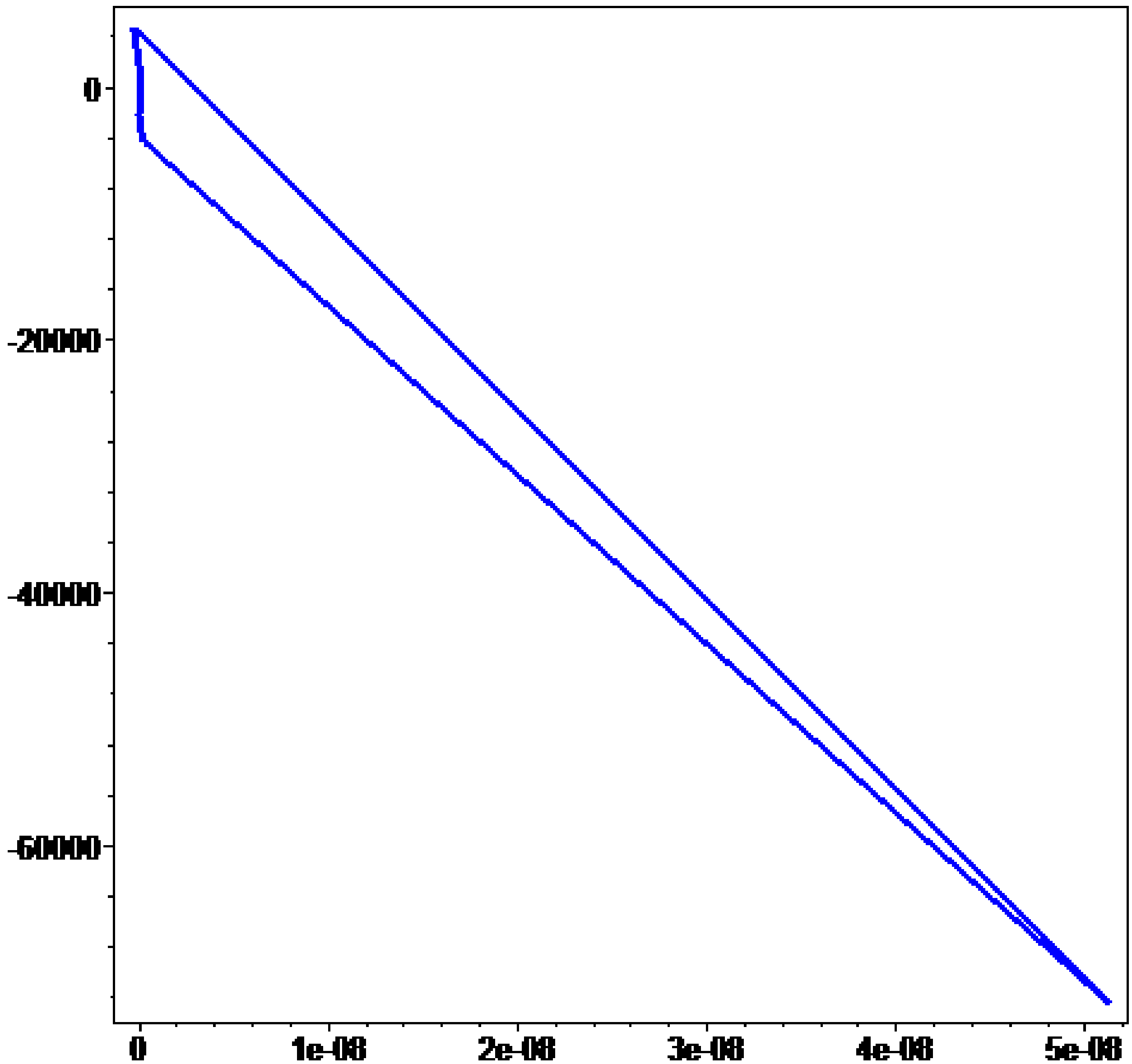
$$e^{(\phi\,I)}-1$$

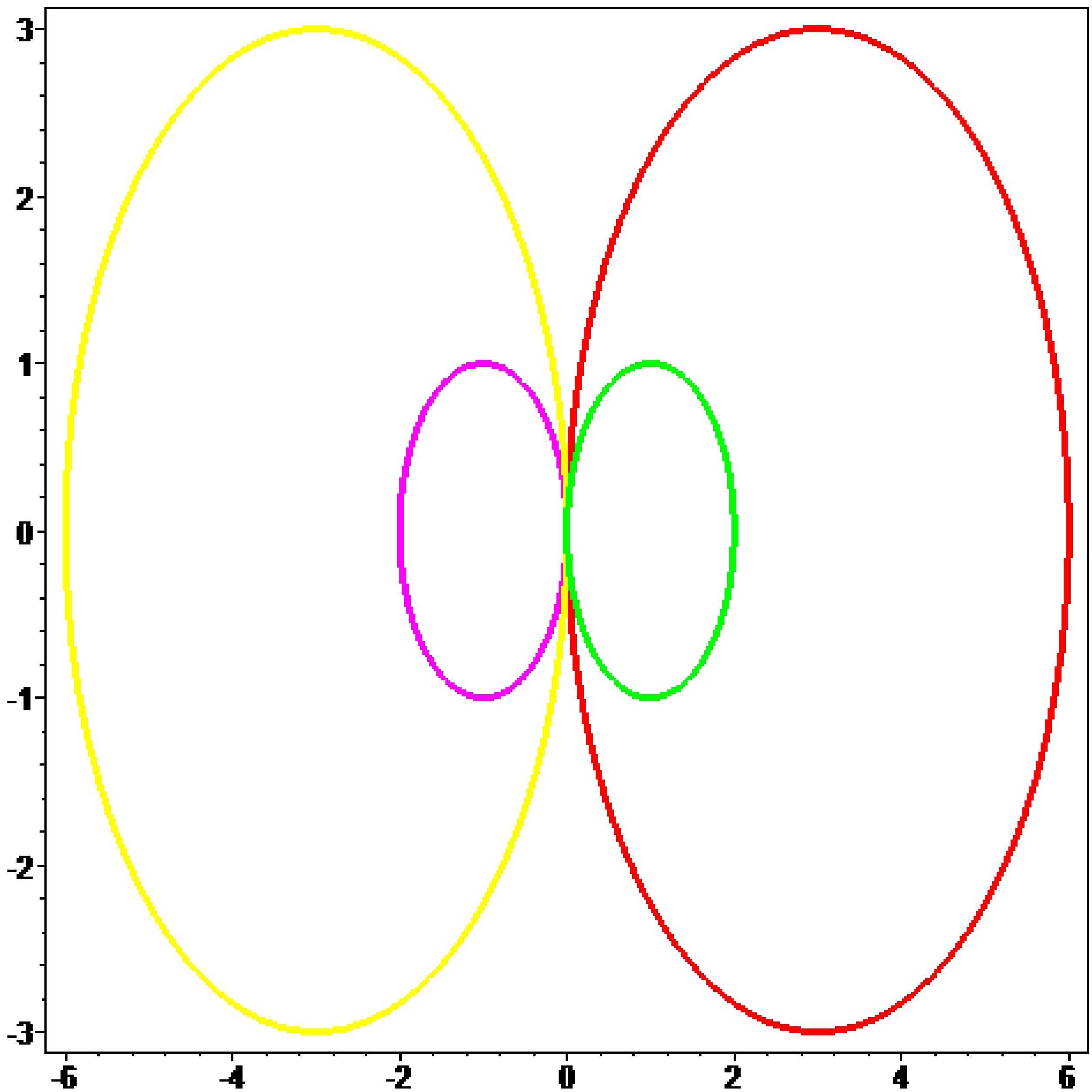
$$\frac{e^{(\phi\,I)}-1}{e^{(\phi\,I)}}$$

$$-\frac{e^{(\phi\,I)}-1}{-\frac{1}{2}e^{(\phi\,I)} - \frac{1}{2}}$$

```
> with(plots):
p1:=complexplot(zz2(0),phi=0..2*Pi,thickness=3,axes=boxed,color=
green):
```

```
p2:=complexplot(zz2(1/3),phi=0..2*Pi,thickness=3,axes=boxed,colo
r=red):
p3:=complexplot(zz2(1/2),phi=0..2*Pi,thickness=3,axes=boxed,colo
r=blue):
p4:=complexplot(zz2(2/3),phi=0..2*Pi,thickness=3,axes=boxed,colo
r=yellow):
p5:=complexplot(zz2(1),phi=0..2*Pi,thickness=3,axes=boxed,color=
magenta):
#display({p1});
#display({p2});
display({p3});
#display({p4});
#display({p5});
display({p1,p2,p4,p5});
```


0
-20000
-40000
-60000
0
1e-08
2e-08
3e-08
4e-08
5e-08




```
>
>
>
> ######################### P4 Generic ######################
> restart;
#y1=y0+O*f0+(1-O)*f1
makeJac:=proc(f)
local N,jac;
N:=LinearAlgebra:-Dimension(f);
jac:=VectorCalculus:-Jacobian(f,[seq(y[i],i=1..N)]);
unapply(jac,y);
end proc;
```

```
makeJacN:=proc(f)
local N,jac;
N:=LinearAlgebra:-Dimension(f);
jac:=VectorCalculus:-Jacobian(f,[seq(y[i],i=1..N)]);
end proc;

Newton:=proc(f,y0,makeJac,tol,maxiter)
#global
local N,dy,Jac,F,yold,ynew,Err,iter;
N:=LinearAlgebra:-Dimension(f);
F:=unapply(f,y);
Jac:=makeJac(f);
yold:=y0;
dy:=LinearAlgebra:-LinearSolve(-Jac(yold),F(yold));
ynew:=yold+dy;
yold:=ynew;
Err:=10;
iter:=1;
while Err>tol and iter < maxiter do
iter:=iter+1;
dy:=LinearAlgebra:-LinearSolve(-Jac(yold),F(yold));
ynew:=yold+dy;
Err:=LinearAlgebra:-Norm(yold-ynew);
yold:=ynew;
end;
#[Err,ynew,iter];
ynew;
end proc;
```

makeJac := proc(*f*) end proc

local *N*, *jac*;

$N := LinearAlgebra:\text{-}Dimension(f);$

$jac := VectorCalculus:\text{-}Jacobian(f, [\mathrm{seq}(y[i], i = 1 .. N)]);$

$\mathrm{unapply}(jac, y)$

makeJacN := proc(*f*)

local *N*, *jac*;

$N := LinearAlgebra:\text{-}Dimension(f);$

$jac := VectorCalculus:\text{-}Jacobian(f, [\mathrm{seq}(y[i], i = 1 .. N)])$

end proc

$Newton$:= proc(f, $y0$, $makeJac$, tol, $maxiter$)
local N, dy, Jac, F, $yold$, $ynew$, Err, $iter$;
$N := LinearAlgebra$:-$Dimension(f)$;
$F :=$ unapply(f, y);
$Jac := makeJac(f)$;
$yold := y0$;
$dy := LinearAlgebra$:-$LinearSolve(-Jac(yold), F(yold))$;
$ynew := yold + dy$;
$yold := ynew$;
$Err := 10$;
$iter := 1$;
while $tol < Err$ and $iter < maxiter$ do
$iter := iter + 1$;
$dy := LinearAlgebra$:-$LinearSolve(-Jac(yold), F(yold))$;
$ynew := yold + dy$;
$Err := LinearAlgebra$:-$Norm(yold - ynew)$;
$yold := ynew$
end do;
$ynew$
end proc

```
> f:=Vector(2,[-y[1]^2,y[1]^2-y[2]]);
y00:=Vector(2,[1.0,0.0]);
```

$$f := \begin{bmatrix} -y_1^2 \\ y_1^2 - y_2 \end{bmatrix}$$

$$y00 := \begin{bmatrix} 1.0 \\ 0. \end{bmatrix}$$

```
> EulerBDF:=proc(f,y00,tf,N)
local F,YY,h,y0,i;
F:=unapply(h*f+y0-Vector(LinearAlgebra:-
Dimension(f),[seq(y[i],i=1..LinearAlgebra:-
Dimension(f))]),y0,h);
h:=tf/N;
YY[0]:=y00;
YY[1]:=Newton(F(y00,h),y00,makeJac,1e-6,50);#print(YY[1]);
for i from 2 to N do
YY[i]:=Newton(F(YY[i-1],h),YY[i-1],makeJac,1e-6,50);
od;
[seq([i*h,YY[i]],i=0..N)];#for printing all the values
YY[N];
```

```
end proc;
```

$EulerBDF$:= proc(f, $y0$0, tf, N)

local F, YY, h, $y0$, i;

F := unapply($h \times f + y0$ − Vector($LinearAlgebra$:-$Dimension$(f),
[seq($y[i]$, $i = 1$ .. $LinearAlgebra$:-$Dimension$(f))]), $y0$, h);

h := tf/N;

$YY[0]$:= $y00$;

$YY[1]$:= Newton(F($y00$, h), $y00$, $makeJac$, 0.1*10^(-5), 50);

for i from 2 to N do

$YY[i]$:= Newton(F($YY[i-1]$, h), $YY[i-1]$, $makeJac$, 0.1*10^(-5), 50)

end do;

[seq([$i \times h$, $YY[i]$], $i = 0$ .. N)];

$YY[N]$

end proc

```
> EulerBDF(f,y00,1,10); # used for comparison
```

$$\begin{bmatrix} 0.516493908066555 \\ 0.268870124077736 \end{bmatrix}$$

```
> ThetaMethod:=proc(f,y00,tf,N,O)
local F,YY,h,y0,i;
ff:=unapply(f,y);
F:=unapply(hh*(OO*ff(y0)+(1-OO)*f)+y0-Vector(LinearAlgebra:-
Dimension(f),[seq(y[i],i=1..LinearAlgebra:-
Dimension(f))]),y0,hh,OO);
h:=tf/N;
YY[0]:=y00;
YY[1]:=Newton(F(y00,h,O),y00,makeJac,1e-6,50);#print(YY[1]);
for i from 2 to N do
YY[i]:=Newton(F(YY[i-1],h,O),YY[i-1],makeJac,1e-6,50);
od;
#[seq([i*h,YY[i]],i=0..N)];#for printing all the values
YY[N];
end proc;
Warning, `ff` is implicitly declared local to procedure `ThetaMethod`
```

$ThetaMethod$:= proc(f, $y00$, tf, N, O)
local F, YY, h, $y0$, i, ff;
ff := unapply(f, y);
F := unapply($hh \times (OO \times ff(y0) + (1 - OO) \times f) + y0$ − Vector(
$LinearAlgebra$:-$Dimension(f)$,
[seq($y[i]$, $i = 1$ .. $LinearAlgebra$:-$Dimension(f)$)]), $y0$, hh, OO);
$h := tf/N$;
$YY[0] := y00$;
$YY[1]$:= Newton($F(y00, h$, O), $y00$, $makeJac$, 0.1*10^(-5), 50);
for i from 2 to N do
$YY[i]$:= Newton($F(YY[i-1], h$, O), $YY[i-1]$, $makeJac$, 0.1*10^(-5), 50)
end do;
$YY[N]$
end proc

```
> ff:=unapply(f,y);
YY[0]:=y00;
q:=h*(O*ff(y0)+(1-O)*f)+y0-Vector(LinearAlgebra:-
Dimension(f),[seq(y[i],i=1..LinearAlgebra:-Dimension(f))]);
F:=unapply(hh*(O*ff(y0)+(1-O)*f)+y0-Vector(LinearAlgebra:-
Dimension(f),[seq(y[i],i=1..LinearAlgebra:-
Dimension(f))]),y0,hh,O);

h:=1/10;
YY[1]:=Newton(F(y00,h,0.5),y00,makeJac,1e-6,50);
YY[2]:=Newton(F(YY[1],h,0.5),YY[1],makeJac,1e-6,50);
>
>
>
>
>
```

$$ff := y \rightarrow \mathrm{rtable}(1..2, \{1 = -y_1^2, 2 = y_1^2 - y_2\}, datatype = anything, subtype = Vector_{column}, storage = rectangular, order = Fortran\_order)$$

$$YY_0 := \begin{bmatrix} 1.0 \\ 0. \end{bmatrix}$$

$$q := y0 + \begin{bmatrix} h\,(-\mathrm{O}\,y0_1^{\,2} - (1-\mathrm{O})\,y_1^{\,2}) - y_1 \\ h\,(\mathrm{O}\,(y0_1^{\,2} - y0_2) + (1-\mathrm{O})\,(y_1^{\,2} - y_2)) - y_2 \end{bmatrix}$$

$$F := (y0, hh, \mathrm{O}) \rightarrow y0 + \mathrm{rtable}(1 .. 2, \{1 = hh\,(-\mathrm{O}\,y0_1{}^2 - (1-\mathrm{O})\,y_1{}^2) - y_1,$$
$$2 = hh\,(\mathrm{O}\,(y0_1{}^2 - y0_2) + (1-\mathrm{O})\,(y_1{}^2 - y_2)) - y_2\}, \mathit{datatype} = \mathit{anything},$$
$$\mathit{subtype} = \mathit{Vector}_{\mathit{column}}, \mathit{storage} = \mathit{rectangular}, \mathit{order} = \mathit{Fortran\_order})$$

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

$$YY_1 := \begin{bmatrix} 0.908712114635714 \\ 0.0869408432040815 \end{bmatrix}$$

$$YY_2 := \begin{bmatrix} 0.832750554934263 \\ 0.151005105471742 \end{bmatrix}$$

```
> i:=0.1;
ThetaMethod(f,y00,1,50,i);
i:=1/3;
ThetaMethod(f,y00,1,50,i);
i:=1/2;
ThetaMethod(f,y00,1,50,i);
i:=2/3;
ThetaMethod(f,y00,1,50,i);
i:=1.0;
ThetaMethod(f,y00,1,50,i);
```

$$i := 0.1$$

$$\begin{bmatrix} 0.502741192105351 \\ 0.279725647282292 \end{bmatrix}$$

$$i := \frac{1}{3}$$

$$\begin{bmatrix} 0.501129248978697 \\ 0.281000909699360 \end{bmatrix}$$

$$i := \frac{1}{2}$$

$$\begin{bmatrix} 0.499974997083022 \\ 0.281912177403608 \end{bmatrix}$$

$$i := \frac{2}{3}$$

$$\begin{bmatrix} 0.498818380304596 \\ 0.282823666431601 \end{bmatrix}$$

$$i := 1.0$$

$$\begin{bmatrix} 0.496498130899834 \\ 0.284647004665424 \end{bmatrix}$$

>
>