Computergestuetzte Mathematik zur Analysis

Lektion 8 (10. Dez.)

```
> restart:
```

Einige Integrale

```
> a:= Int((1-t^4)^(-1/2),t=0..1); # Betafunktion
> value(a);
> b:= Int(t^2*(1-t^2)^(-1/2),t=0..1);
> value(b);
> A := Int(exp(-x^2), x = -infinity .. infinity);
```

A) 1

B) $\sqrt{\pi}$

```
> value(A);
> B := Int(exp(-x^2), x);
> value(B);
> C := Int(ln(x)^2 / (1 + x)^2, x);
> value(C);
> E := Int(ln(x)^2 / (1 + x)^2, x = 0 .. infinity);
> value(E);
```

A := evalf(E, 15);

B := evalf(value(E),15):

1) A und B stimme uebeerein

2) A ist genauer als B

3) B ist genauer als A

```
> evalf(E, 15);
> evalf(value(E), 15);
> evalf(value(E),20);
  evalf(E,20);
```

Riemann Integral

```
> restart:
> S := Sum(a/n*exp(k*a/n), k = 0 .. n-1);
> value(S);
> L := Limit(S, n = infinity);
> value(L);
```

Grenzwerte

```
> restart:
> a := k^2/(2^k);
```

1) Die Folge der a_k konvergiert gegen 0

2) Die Folger der a_k divergiert

3) Die Folge der a_k konvergiert gegen $\sqrt{\pi}$

```
> A := Limit(a, k = infinity);
> value(A);
> b := k^k/k!;
```

1) Die Folge der b_k konvergiert gegen 0

2) Die Folge der b_k divergiert

3) Die Folge der b_k konvergiert gegen $\sqrt{2}$

```
> B := Limit(b, k = infinity);
> value(B);
> c := (-1)^k*b;
> C := Limit(c, k = infinity);
```

```
> value(C);
> e := sin(k*Pi);
> E := Limit(e, k = infinity);
> value(E);
> value(E) assuming k::integer;
> e assuming k::integer;
```

Reihen

```
> restart:
> a := 1/k/(k+1);
> A := Sum(a, k = 1 .. infinity);
> value(A);
> A1 := Sum(a, k = 1 .. N);
> value(A1);
> b := 1/k^3;
> B := Sum(b, k = 1 .. infinity);
> value(B);
> evalf(%);
```

Produkte

```
> p:= Product(1-1/k^2,k=2..infinity);
> value(p);
> p:= Product(1-1/(2*k)^2,k=1..infinity); # Wallis Formel
> value(p);
> product(GAMMA(k/3),k=1..8);
```

Gleichmaessige Konvergenz

```
> a := (4*sin(x)*(1/5))^k;

> S := Sum(a, k = 1 .. infinity);
> f := value(S);
> farben := [black, red, yellow, blue, green, magenta, coral,
  pink, cyan];
> funktionen := [f, seq(sum(a, k = 1 .. n), n = 1 .. 7)];
> plot(funktionen, x = -Pi .. Pi, color = farben);
> funktionen := [f, f+1/3, f-1/3, seq(sum(a, k = 1 .. 5*n), n = 1
  .. 4)]:
> farben := [black, gray, gray, red, yellow, blue, green,
  magenta, coral, pink, cyan];
> plot(funktionen, x = -Pi .. Pi, color = farben, thickness = 2,
  numpoints = 500);
```

Das Taylorpolynom

```
> f := sqrt(1+x)/sqrt(1+x^2);
> t := series(f, x  = 0, 8);
> P := convert(t, polynom);
> for n from 1 to 3 do;
>   t := series(f, x = 0, n + 1);
>   P[n] := convert(t, polynom);
> od;
> P[0] := f;
> farbe := [black,blue, red, cyan];
> plot(convert(P, list), x = -.8 .. .8, color = farbe, thickness
  = 2);
> g := cos(x);
> for n in [4, 20, 60] do;
>   Q[n] := convert(series(g, x, n+1), polynom):
    E[n] := Q[n] - g;
> od:
> Q[4];
> E[0] := 0;
> Q[0] := g;
> plot(convert(Q, list), x = - 32 .. 32, y = -2 .. 2, color =
  farbe, thickness = 2, numpoints = 1000);
```

Das komplexe Bild

```
> restart:
> h := arctan(x);
> for n in [4, 20, 60] do;
>   S[n] := convert(series(h, x = 0, n+1), polynom):
> od:
> S[0] := h;
> plot(convert(S, list), x = -2 .. 2, -2 .. 2, thickness = 2,
  numpoints = 500);
> plot([h, S[60]], x = -2 .. 2, -2 .. 2, color = [blue, green],
  thickness = 2, numpoints = 500);
> x := a + I*b;
> pl1 := plot3d(abs(h-S[20]), a = -1.3 .. 1.3, b = -1.3 .. 1.3,
  shading = zhue,  style = patchcontour, numpoints = 2000):
> with(plots):
> display(pl1, axes = boxed, view = 0 .. 5, orientation = [-110,
  35]);
> pl2 := plot3d(abs(h), a = -1.3 .. 1.3, b = -1.3 .. 1.3, color =
  black,  style = wireframe, numpoints = 6000):
> display([pl1, pl2], axes = boxed, view = 0 .. 2, orientation =
```

```
   [-110, 35]);
> x := 'x':
> plot(abs(arctan(I*x)),x=-2..2);
```