# Computergestuetzte Mathematik zur Analysis

## Lektion 6 (25. Nov.)

## Loesen von Gleichungen (solve / fsolve)

```
> Glg := (x-1)^2 = 4-x;
> Lsg := solve(Glg, x);
> subs(x = Lsg[1], Glg);
> subs(x = Lsg[2], Glg);
> evalb((-1/2-1/2*sqrt(13))^2=7/2+1/2*sqrt(13));
> evalb((1/4+1/2*sqrt(13)+13/4)=7/2+1/2*sqrt(13));
> evalb(expand((-1/2-1/2*sqrt(13))^2)=7/2+1/2*sqrt(13));
> simplify(op(1,??)-op(2,??));
> Gls := {x^2 + y^2 = 1, x = y};
> vars := {x, y};
> Lsg := solve(Gls, vars);
> allvalues(Lsg);
> Glg;
> plot([rhs(Glg),lhs(Glg)],x=-2..3,color=[red,blue]);
> solve(Glg);
> FLsg := fsolve(Glg,x);
> subs(x = FLsg[1], Glg);
> FLsg := fsolve(Gls, vars);
> FLsgA := fsolve(Gls, vars, avoid = {FLsg});
> solve(sin(x) = cos(x), x);
> plot([sin,cos],-5..5,color=[cyan,magenta]);
> _EnvAllSolutions := true;  #Umgebungsvariable
> solve(sin(x) = cos(x), x);
> about(_Z1);
> _EnvAllSolutions := false;
> id := x -> x;
> plot([id, tan], -6 .. 18, -6 .. 18, discont = true, thickness =
  2,color=[red,blue]);
> Glg := tan(x) = x;
> solve(Glg, x);
> fsolve(Glg, x);
> fsolve(Glg, x, avoid = {x = 0});
> fsolve(Glg, x = 4 .. 6);
> f := x -> 2^x;
```

```
> g := x -> x^2;
> plot([f, g], -2 .. 4.5, thickness = 2,color=[red,blue]);
> Glg := f(x) = g(x);
> solve(Glg, x);
> evalf(%);
```

## Graphen von Loesungsmengen

```
> with(plots):
> Gls := {x^2+y^2=1, x=y};
> implicitplot(Gls, x = -1 .. 1, y = -1 .. 1, thickness = 2,
  scaling = constrained);
> Glg := x*y = 0;
> implicitplot(Glg, x = -1 .. 1, y = -1 .. 1, thickness = 2, axes
  = frame);
> Glg := x^2*y = 0;
> implicitplot(Glg, x = -1 .. 1, y = -1 .. 1, thickness = 2, axes
  = frame);
> implicitplot(Glg, x = -1 .. 1, y = -1 .. 1, thickness = 2, axes
  = frame, scaling= constrained, gridrefine=3);
> Glg := (x^2 + y^2)^2 + 3*x^2*y - y^3;
> implicitplot(Glg, x = -1 .. 1, y = -1 .. 1, thickness = 2, axes
  = frame, scaling = constrained);
> implicitplot(Glg, x = -1 .. 1, y = -1 .. 1, thickness = 2, axes
  = frame, scaling = constrained,numpoints=60000);
> P1 := (x^2 + y^2)^3 - 4*x^2*y^2;
> implicitplot(P1, x = -.8 .. .8, y = -.8 .. .8, numpoints =
  20000, scaling = constrained, thickness = 2, axes = boxed);
> with(algcurves):
> plot_real_curve(P1, x , y, thickness = 2);
> plot1 := plot3d(P1, x = -.2 .. .2, y = -.2 .. .2, shading =
  zhue, numpoints = 60000, style = patchnogrid):
> plot1;
> plot2 := plot3d(-0, x = -.2 .. .2, y = -.2 .. .2, color =
  black, style = patchnogrid,view=-0.01..0.01):
> plot2
> display({plot1, plot2}, axes = boxed);
```

## Wurzeln von Polynomen

```
> restart;n := 3; a := -1/2; b := 3;
```

$$> f := \frac{1}{2^n \cdot n!}(1-x)^{-a}(1+x)^{-b}\left(\frac{\partial^n}{\partial x^n}\left((1-x)^a(1+x)^b(1-x^2)^n\right)\right);$$

*#Jacobi Polynom*

```
> simplify(%);
```

```
> plot(f, x = -1 .. 1.5, -2 .. 4, thickness = 2);
> solve(f = 0);
> Lsg := [%];
> nops(Lsg);
> map(Im, Lsg);
> r := simplify(%);
> map(Re, Lsg);
> simplify(%);
> g := x^7 - 3*x^6 + 2*x^5 + x^3 + 4*x^2 - 19*x + 14;
> plot(g, x = -2 .. 2.7, -50 .. 50, thickness = 2);
> Lsg := solve(g = 0);
> allvalues([Lsg]);
> fsolve(g = 0);
> num_Lsg := fsolve(g = 0, x, complex);
> for z in num_Lsg do
>   z;
> od;
```

# Ersetzungen

```
> restart;
> r := (a*x^2 + b*x + c)^3;
> subs(a = 1, b = -1, c = 3, x = 0, r);
> r;
```

Bestimme den geraden Anteil von r

```
> 1/2*(r + subs(x = -x, r));
> g := expand(%);
> cg := collect(g, x);
> subs(x^2 = y, cg);
> algsubs(x^2 = y, cg);
> collect(%, y);
```

Subs macht manchmal Fehler:

```
> h := x/sin(Pi*x);
> subs(x = 0, h);
> plot(h, x = -0.1e-3 .. 0.1e-3);
> limit(h, x = 0);
> a := cos(x+y);
> a = expand(a);
> b := sin(x-y):
> b = expand(b);
> A := cos(x)*cos(y);
> A = combine(A);
> c:= Int(sin(x),x=1..2);
```

```
> d:= Int(cos(x),x=1..2);
> combine(c+d);
> expand(sin(x+y));
> trigsubs(sin(x+y));
> trigsubs(sin(2*z) = 2*cos(z)*sin(z), sin(2*z)*cos(z));
```

## Vereinfachungen / Annahmen

```
> restart;
> simplify(exp(x^2+ln(c*exp(y^2))-x^2));
> simplify(sin(x)^2+ln(2*x)+cos(x)^2,trig);
> simplify(sqrt(x^2), assume = positive);
> g := int(x^2*(exp(x)+exp(-x)), x);
> collect(g, exp);
> collect(g, x);
> normal((x^2-y^2)/(x+y)^2);
> exint := int(exp(a*t), t = 0 .. infinity);
  assume(a < 0);
  exint;
> about(a);
> additionally(a > -2);
> about(a);
> e := ln(y/x)-ln(y)+ln(x);
> simplify(e);
> simplify(e) assuming y::positive;
> simplify(e) assuming y::positive, x::positive;
> about(x);
```

### Beispiel

Weitere Vereinfachungen

```
> restart;
> F := tan(x)^2 + 1;
> simplify(F);
> convert(F, sin);
> convert(F,exp);
> G := tan(3*x);
> G = expand(G);
> H := tan(x) + tan(y);
> convert(H, sincos);
> normal(??);
> H1:=combine(??);
> zaehler := numer(H1); nenner := denom(H1);
```

```
> H = combine(zaehler) / nenner;
```

## Maple rechnet komplex

```
> I^2;
> sqrt(-4);
> z := 1 + 3*I;
> Re(z);
> Im(z);
> conjugate(z);
> abs(z);
> z := x + I*y;
> Re(z);
```

```
> Re(z) assuming x::real, y::real;
> abs(z);
> abs(z) assuming x::real, y::real;
> evalc(abs(z));
> evalc(sin(x+I*y));
```

## auch wenn man nicht damit rechnet

```
> f := 1/x;
> F := int(f, x);
> int(f, x = -2 .. -1);
> subs(x = -2, F);
> evalf(%);
> simplify(ln(-2));
> Nl := ln(x + I*y);
> r := Re(Nl);
> i := Im(Nl);
> plot3d(r, x = -2 .. 2, y = -2 .. 2, shading = zhue, axes =
  boxed);
> plot3d(i, x = -2 .. 2, y = -2 .. 2, shading = zhue, axes =
  boxed, orientation = [-113, 37]);
>
```