Blatt 10

Aufgabe 39

```
> restart;
> with(LinearAlgebra):
> with(VectorCalculus):
> phi := (x1, x2) -> < x2, cos(x1) * cosh(x2), sin(x1) * cosh(x2)
  >;
> Dphi[1] := diff(phi(x1, x2), x1);
> Dphi[2] := diff(phi(x1, x2), x2);
> G := Matrix(2):
> for i from 1 to 2 do
    for j from 1 to 2 do
      G(i, j) := simplify(Dphi[i] . Dphi[j]);
    end do;
  end do;
> G;
> Determinant(G);
```

Aufgabe 40

```
> restart;
> with(plots):
> # Seiten des Einheitsquadrates
> Q := <t, -1>, <t, 1>, < -1, t>, < 1, t>;
> # Gegebene Matrix
> A := < < 3 | 1/2 >, < 1/2 | 2 > >;
> # Transformierte Seiten
> images := seq(A . q, q in Q);
> p1 := plot([ seq([ op(convert(Q[kk], list)), t = -1..1 ], kk =
  1..4) ], color = blue, legend = [ 'Urbild', 'Urbild', 'Urbild',
  'Urbild' ]);
> p2 := plot([ seq([ op(convert(images[kk], list)), t = -1..1 ],
  kk = 1..4) ], color = red, legend = [ 'Bild', 'Bild', 'Bild',
  'Bild' ]);
> display(p1, p2);
```

Aufgabe 41

```
> restart:
> with(plots):
> f := (x, y) -> (3*x^2 + x + y - 3*y^2) * exp(-(x^2 + y^2));
> schnitte_x := [ x, y, f(x, y), y = -2..2 ];
> schnitte_y := [ x, y, f(x, y), x = -2..2 ];
> schnitte_xy  := [ x, x+y, f(x, x+y), x = max(-2, -2-y)..min(2,
  2-y) ];
> schnitte_xmy := [ x, -x+y, f(x, -x+y), x = max(-2, y-2)..min(2,
  y+2) ];
> p1 := plot3d(f(x, y), x = -2..2, y = -2..2, style =
  surfacecontour);
> p2 := spacecurve({ seq(schnitte_x, x = -2..2, 1/2) }, color =
  black, thickness = 2):
> p3 := spacecurve({ seq(schnitte_y, y = -2..2, 1/2) }, color =
  black, thickness = 2):
> p4 := spacecurve({ seq(schnitte_xy, y = -3.5..3.5, 0.5) },
  color = black, thickness = 2):
> p5 := spacecurve({ seq(schnitte_xmy, y = -3.5..3.5, 0.5) },
  color = black, thickness = 2):
> display(p1, p2, p3);
> display(p1, p4, p5);
```

Aufgabe 42

```
> restart;
> with(LinearAlgebra):
> f := (x, y, z) -> x^2 - y^2 + z^2 - (x^2 + 2*y^2 +4*z^2)^2;
> Df := [ diff(f(x, y, z), x), diff(f(x, y, z), y), diff(f(x, y,
  z), z) ];
> D2f := < < diff(f(x, y, z), x$2), diff(f(x, y, z), [ x, y ]),
  diff(f(x, y, z), [ x, z ]) > |
  < diff(f(x, y, z), [ y, x ]), diff(f(x, y, z), y$2), diff(f(x,
  y, z), [ y, z ]) > |
  < diff(f(x, y, z), [ z, x ]), diff(f(x, y, z), [ z, y ]), diff
  (f(x, y, z), z$2) > >;
> kritischePunkte := solve([ seq(Df[i] = 0, i = 1..3) ], [ x, y,
  z ]):
> kritischePunkte := seq(allvalues(kritischePunkte[kk]), kk = 1..
  nops(kritischePunkte));
> # Reelle Lösungen
> kritischePunkte := [ seq(kritischePunkte[kk], kk in [ 1, 2, 3,
  6, 7 ]) ];
> # Prüfe Definitheit der Hesse-Matrix
> seq(print(kritischePunkte[kk], 'EW' = Eigenvalues(subs
  (kritischePunkte[kk], D2f))), kk = 1..nops(kritischePunkte));
> # Das heißt: 0 und (0,0,+-sqrt(2)/8) sind Sattelpunkte,
  # die anderen beiden lokale Maxima
> plot3d(f(x, 0, y), x = -1..1, y = -1/2..1/2, style =
  patchcontour, contours=35, view = -0.3 .. 0.3, numpoints =
  3000);
> plot3d(f(x, y, 0), x = -1..1, y = -1/2..1/2, style =
  patchcontour, contours=35, view = -0.3 .. 0.3, numpoints =
3000);
```