

```
In[1]:= Clear["Global`*"];
```

Problem

Solve numerically the differential equation

$$\frac{\partial u(x, y, t)}{\partial t} = \frac{\partial^2 u(x, y, t)}{\partial x^2} + \frac{\partial^2 u(x, y, t)}{\partial y^2} \quad (1)$$

with the following initial condition

$$u(x, y, 0) = e^{-(x^2+y^2)} \quad (2)$$

and Dirichlet boundary conditions

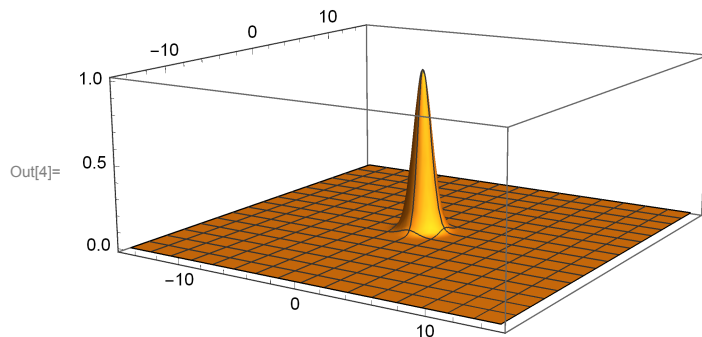
$$u(x, y, t) = 0 \text{ for at least one of } x \text{ and } y \text{ going to infinity.} \quad (3)$$

Exact solution

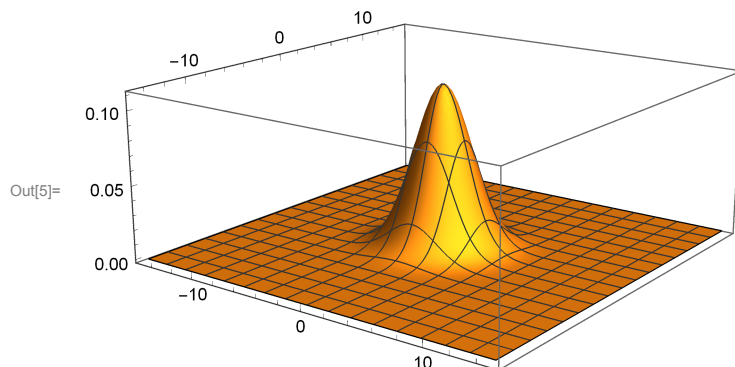
```
In[2]:= u0[x_, y_, t_] = Exp[-(x^2 + y^2) / (1 + 4 t)] / (1 + 4 t);
sol = DSolve[{D[u[x, y, t], t] == D[u[x, y, t], x, x] + D[u[x, y, t], y, y],
  u[x, y, 0] == u0[x, y, 0]}, u, {x, y, t}]
```

```
Out[3]= DSolve[
  {u^(0,0,1)[x, y, t] == u^(0,2,0)[x, y, t] + u^(2,0,0)[x, y, t], u[x, y, 0] == e^{-x^2-y^2}}, u, {x, y, t}]
```

```
In[4]:= Plot3D[u0[x, y, 0], {x, -15, 15}, {y, -15, 15}, PlotRange -> All, PlotPoints -> 100]
```



```
In[5]:= Plot3D[u0[x, y, 2], {x, -15, 15}, {y, -15, 15}, PlotRange -> All, PlotPoints -> 100]
```



Numerical solution using built-in function

```
In[71]:= {xmin, xmax} = {-15, 15};
{ymin, ymax} = {-15, 15};
{tmin, tmax} = {0, 10};
numsol = NDSolve[{D[u[x, y, t], t] == D[u[x, y, t], x, x] + D[u[x, y, t], y, y],
  u[x, y, 0] == u0[x, y, 0], u[xmin, y, t] == u0[xmin, y, 0], u[xmax, y, t] == u0[xmax, y, 0],
  u[x, ymin, t] == u0[x, ymin, 0], u[x, ymax, t] == u0[x, ymax, 0]},
  u, {x, xmin, xmax}, {y, ymin, ymax}, {t, tmin, tmax}]
```

... **NDSolve**: Using maximum number of grid points 100 allowed by the MaxPoints or MinStepSize options for independent variable x.

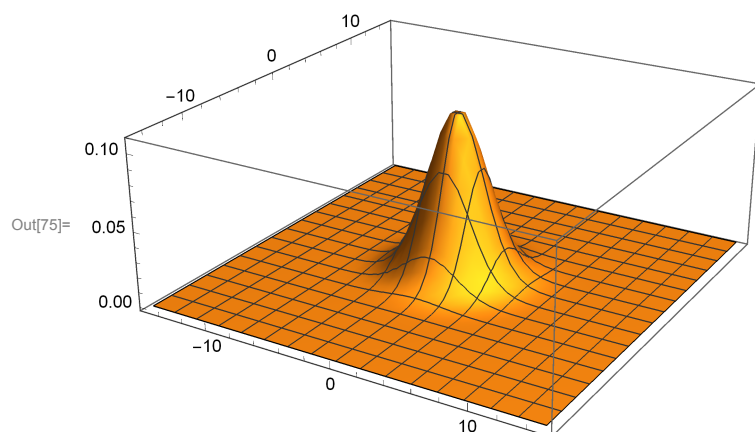
... **NDSolve**: Using maximum number of grid points 100 allowed by the MaxPoints or MinStepSize options for independent variable y.

... **NDSolve**: Using maximum number of grid points 100 allowed by the MaxPoints or MinStepSize options for independent variable x.

... **General**: Further output of NDSolve::mxsst will be suppressed during this calculation.

```
Out[74]= {{u -> InterpolatingFunction[  Domain: {{-15., 15.}, {-15., 15.}, {0., 10.}}
Output: scalar ] ]}}
```

```
In[75]:= Plot3D[Evaluate[(u[x, y, t] /. numsol) /. {t -> 2}],
  {x, xmin, xmax}, {y, ymin, ymax}, PlotRange -> All, PlotPoints -> 40]
```



Numerical solution using basic explicit methods

```

In[50]:= (* grid initialization *)
{xmin, xmax} = {-15, 15};
{ymin, ymax} = {-15, 15};
{tmin, tmax} = {0, 5};
{nx, ny, nt} = {51, 51, 51};
dx = (xmax - xmin) / (nx - 1);
dy = (ymax - ymin) / (ny - 1);
dt = (tmax - tmin) / (nt - 1);
Print["λ = ", λ = N[dt / dx^2]]
X = N[Range[xmin, xmax, dx]];
Y = N[Range[ymin, ymax, dy]];
T = N[Range[tmin, tmax, dt]];

(* Initialization of the array with zeroes - Dirichlet's boundary conditions *)
v = ConstantArray[0.0, {nx, ny, nt}];
error = ConstantArray[0.0, nt];

(* Initial state *)
Do[v[[i, j, 1]] = u0[X[[i]], Y[[j]], tmin], {j, 2, ny - 1}, {i, 2, nx - 1}];
Do[v[[i, j, 2]] = u0[X[[i]], Y[[j]], tmin + dt], {j, 2, ny - 1}, {i, 2, nx - 1}];

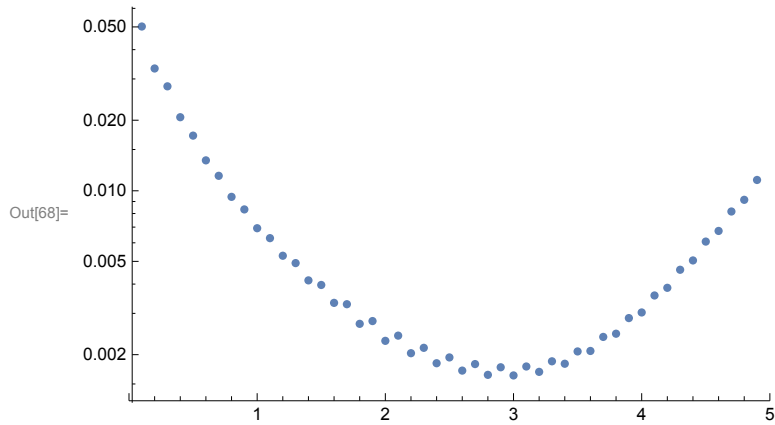
method = 1;
Which[
  method == 1,
  Print["Euler explicit method - order 1, stable for λ = k/h^2 < 1/4"];
  Do[v[[i, j, n + 1]] =
    v[[i, j, n]] + λ (v[[i + 1, j, n]] + v[[i, j + 1, n]] - 4 v[[i, j, n]] + v[[i - 1, j, n]] +
      v[[i, j - 1, n]]), {n, 1, nt - 1}, {j, 2, ny - 1}, {i, 2, nx - 1}]
];
Do[
  error[[n]] = 0.0;
  Do[
    error[[n]] = Max[error[[n]], Abs[v[[i, j, n]] - u0[X[[i]], Y[[j]], T[[n]]]]],
    {j, 2, ny - 1}, {i, 2, nx - 1}
  ],
  {n, 1, nt - 1}
];

λ = 0.2777777778

Euler explicit method - order 1, stable for λ = k/h^2 < 1/4

```

```
In[68]:= (* show maximal errors *)
ListLogPlot[Table[{T[[n]], error[[n]]}, {n, 1, nt}],
  PlotRange → All]
Print["Maximal error: ", Max[error]]
```



Maximal error: 0.05020090754

```
In[70]:= ListPlot3D[Flatten[Table[{X[[i]], Y[[j]], v[[i, j, 50]]}, {j, 1, ny}, {i, 1, nx}], 1],
  PlotRange → All]
```

