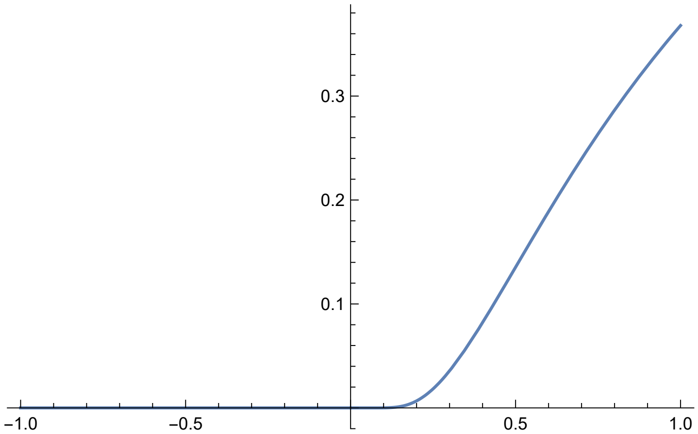
Pensieve header: A demo of the basic C^∞ Lego blocks.

In[∘]:= σ[x_] := $\begin{cases} e^{-1/x} & x > 0 \\ 0 & x \le 0 \end{cases}$

```
Plot[σ[x], {x, -1, 1}]
```

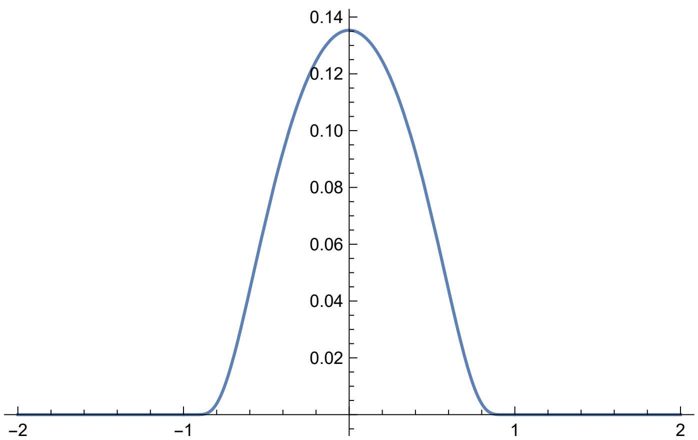
Out[∘]=



In[∘]:= $\beta_{\epsilon\_}$[x_] := σ[ϵ + x] × σ[ϵ − x]

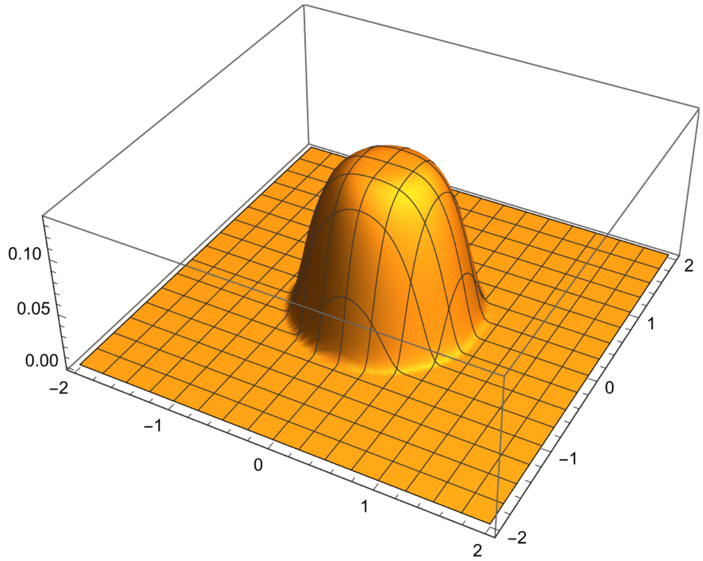
```
Plot[β1[x], {x, -2, 2}]
```

Out[∘]=



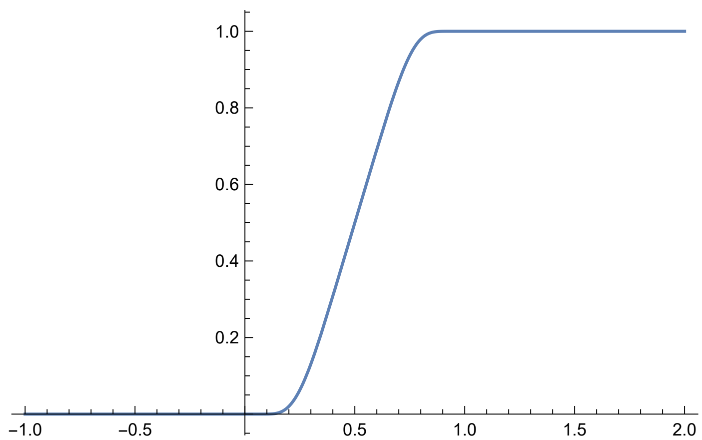
In[∘]:=
```
β_{a_,ϵ_}[z_] := β_{ϵ^2}[Norm[z - a]^2]
Plot3D[β_{0,1}[{x, y}], {x, -2, 2}, {y, -2, 2},
 PlotPoints → 100, PlotRange → All, Exclusions → None]
```

Out[∘]=



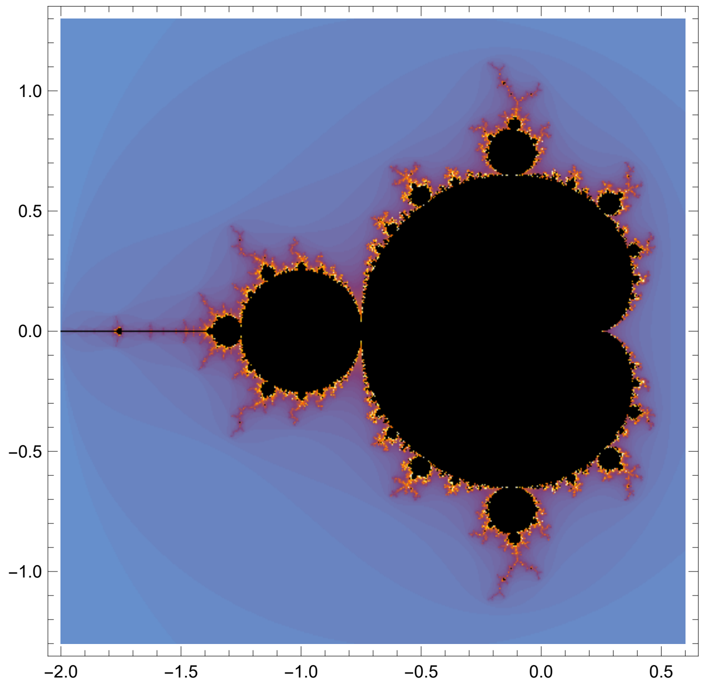
In[∘]:=
```
Z = NIntegrate[β_{1/2,1/2}[t], {t, 0, 1}];
θ[x_] := Z^-1 NIntegrate[β_{1/2,1/2}[t], {t, 0, x}]
Plot[θ[x], {x, -1, 2}]
```

Out[∘]=



In[∘]:=
```
MandelbrotSetPlot[]
```

Out[∘]=



In[∘]:=
```
ϵ = 0.2;
pts = Select[
  Join @@ Table[{x, y}, {x, -2, 0.5, ϵ}, {y, -1, 1, ϵ}],
  MandelbrotSetMemberQ[#[[1]] + ⅈ #[[2]]] &
 ]
```

MandelbrotSetMemberQ: Maximum number of iterations reached. Point may not actually be in the Mandelbrot set, but just extremely close.

MandelbrotSetMemberQ: Maximum number of iterations reached. Point may not actually be in the Mandelbrot set, but just extremely close.

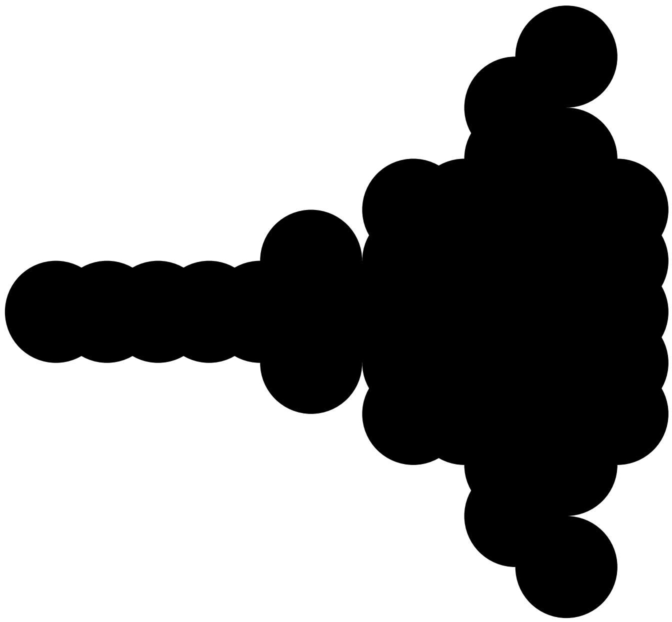
MandelbrotSetMemberQ: Maximum number of iterations reached. Point may not actually be in the Mandelbrot set, but just extremely close.

General: Further output of MandelbrotSetMemberQ::maxiter will be suppressed during this calculation.

Out[∘]=
```
{{-2., 0.}, {-1.8, 0.}, {-1.6, 0.}, {-1.4, 0.}, {-1.2, 0.}, {-1., -0.2},
 {-1., 0.}, {-1., 0.2}, {-0.8, 0.}, {-0.6, -0.4}, {-0.6, -0.2}, {-0.6, 0.},
 {-0.6, 0.2}, {-0.6, 0.4}, {-0.4, -0.4}, {-0.4, -0.2}, {-0.4, 0.}, {-0.4, 0.2},
 {-0.4, 0.4}, {-0.2, -0.8}, {-0.2, -0.6}, {-0.2, -0.4}, {-0.2, -0.2},
 {-0.2, 0.}, {-0.2, 0.2}, {-0.2, 0.4}, {-0.2, 0.6}, {-0.2, 0.8}, {0., -1.},
 {0., -0.6}, {0., -0.4}, {0., -0.2}, {0., 0.}, {0., 0.2}, {0., 0.4}, {0., 0.6},
 {0., 1.}, {0.2, -0.4}, {0.2, -0.2}, {0.2, 0.}, {0.2, 0.2}, {0.2, 0.4}}
```

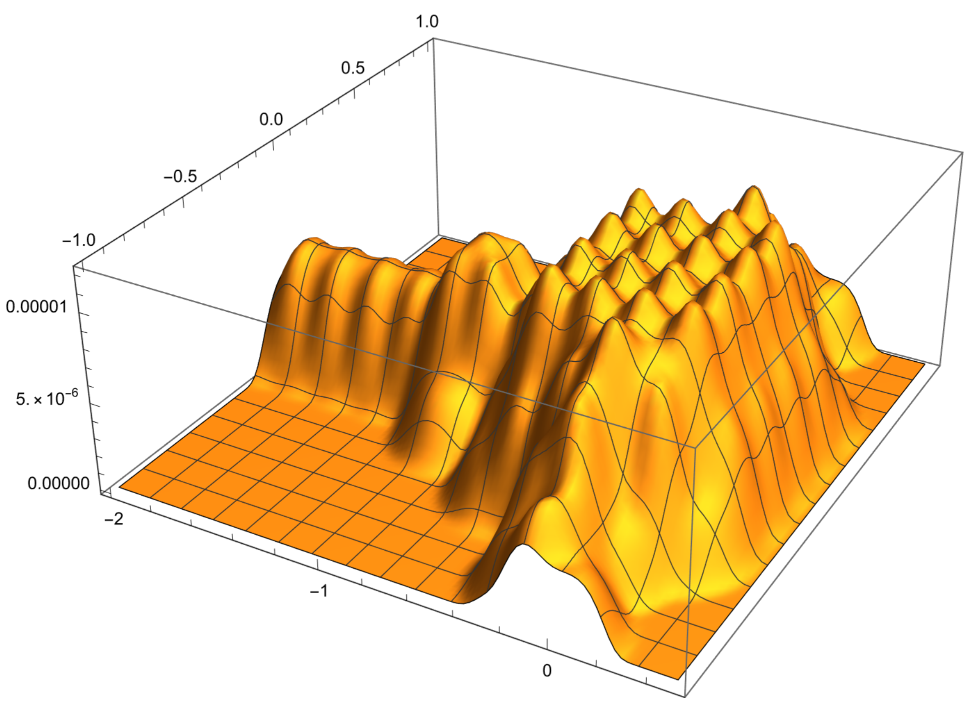
In[∘]:=
```
Graphics[pts /. {x_, y_} :→ Disk[{x, y}, ϵ]]
```

Out[∘]=



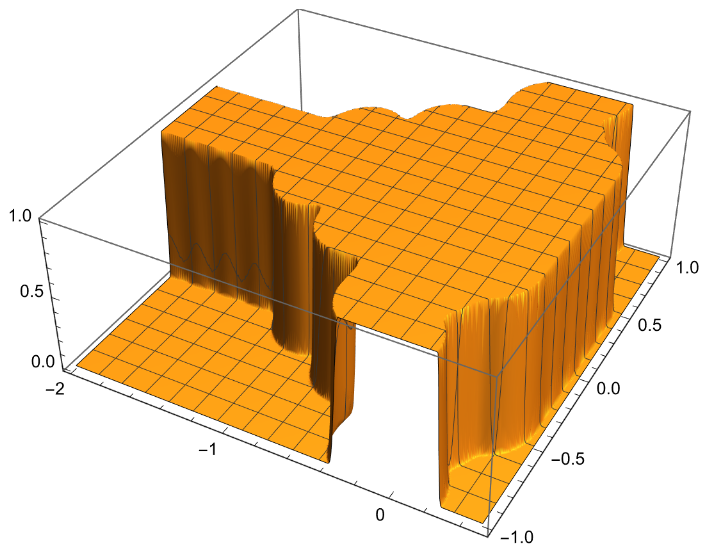
In[∘]:=
```
g[x_, y_] := Sum[β_{p,2ϵ}[{x, y}], {p, pts}];
Plot3D[g[x, y], {x, -2, 0.5}, {y, -1, 1},
 PlotPoints → 100, PlotRange → All, Exclusions → None]
```

Out[∘]=



In[∘]:=
```
f[x_, y_] := θ[10^6 g[x, y]];
Plot3D[f[x, y], {x, -2, 0.5}, {y, -1, 1},
 PlotPoints → 100, PlotRange → All, Exclusions → None]
```

Out[∘]=



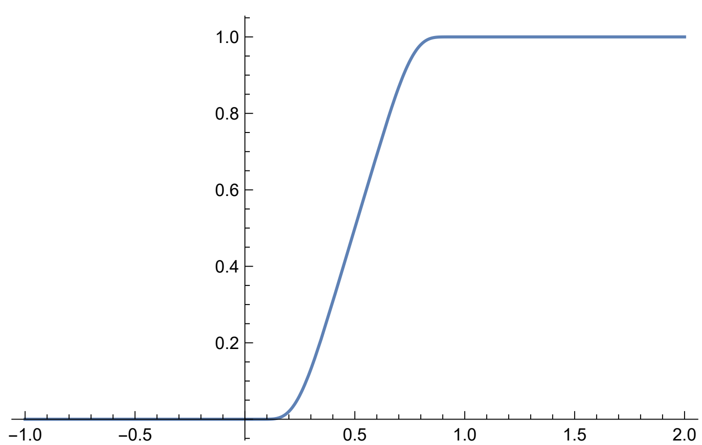
In[∘]:=
```
θ1 = Interpolation[Table[{x, θ[x]}, {x, 0, 1, 0.01}]]
```

Out[∘]= InterpolatingFunction[Domain: {{0., 1.}} Output: scalar]

In[∘]:=
```
Plot[θ1[x], {x, -1, 2}]
```

InterpolatingFunction: Input value {-0.999939} lies outside the range of data in the interpolating function. Extrapolation will be used.

Out[∘]=



In[∘]:=
```
points = {{0, 0}, {1, 1}, {2, 3}, {3, 4}, {4, 3}, {5, 0}};
```

In[ ]:=
```
f1[x_, y_] := θ1[3 × 10^5 g[x, y]];
Plot3D[f1[x, y], {x, -2, 0.5}, {y, -1, 1},
 PlotPoints → 100, PlotRange → All, Exclusions → None]
```

Out[ ]=