Neal's Funnel

Jose Storopoli

Created on 09/03/2021. Updated on 09/03/2021

In this notebook we analyze Neal's Funnel (Neal, 2011). Neal (2011) defines a distribution that exemplifies the difficulties of sampling from some hierarchical models. Neal's example is fairly extreme, but can be trivially reparameterized in such a way as to make sampling straightforward. Neal's example has support for $y \in \mathbb{R}$ and $x \in \mathbb{R}^9$ with density

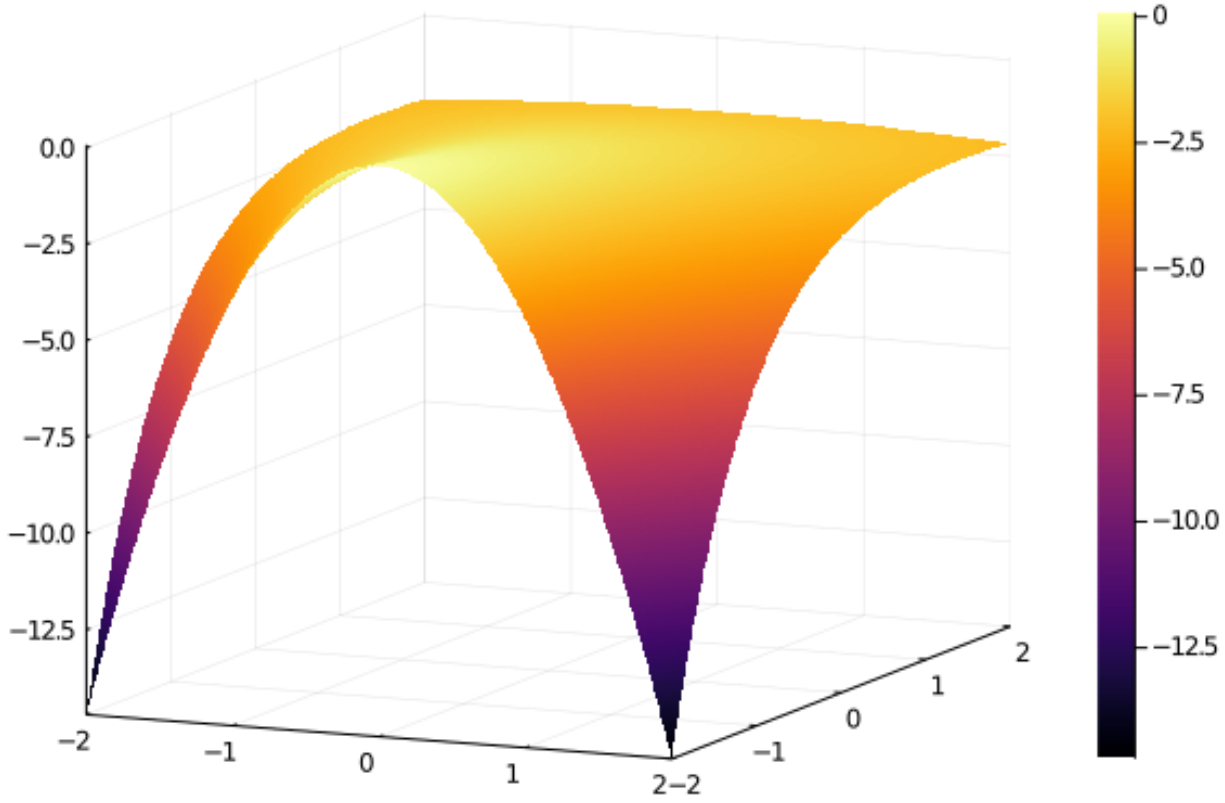
$$p(y, x) = \text{Normal}(y \mid 0, 3) \times \prod_{n=1}^{9} \text{normal}\left(x_n \mid 0, \exp\left(\frac{y}{2}\right)\right).$$

The probability contours are shaped like ten-dimensional funnels. The funnel's neck is particularly sharp because of the exponential function applied to y. I won't try to demonstrate it in 9 dimensions but I will show it in 3 dimensions. Below there is is the Neal's Funnel density in 3-D. This is partially taken from a StanCon 2018 YouTube video by Ben Goodrich[1]

```
using Distributions, Plots

x = -2:0.01:2;
kernel(x, y) = logpdf(Normal(0, exp(y / 2)), x)
surface(x, x, kernel)
```

[1] see from 45' onwards.



So what if we reparameterize so that we can express y and x_n as standard normal distributions, by using a reparameterization trick[2]:

$$x^* \sim \text{Normal}(0, 0)$$
$$x = x^* \cdot \sigma_x + \mu_x$$

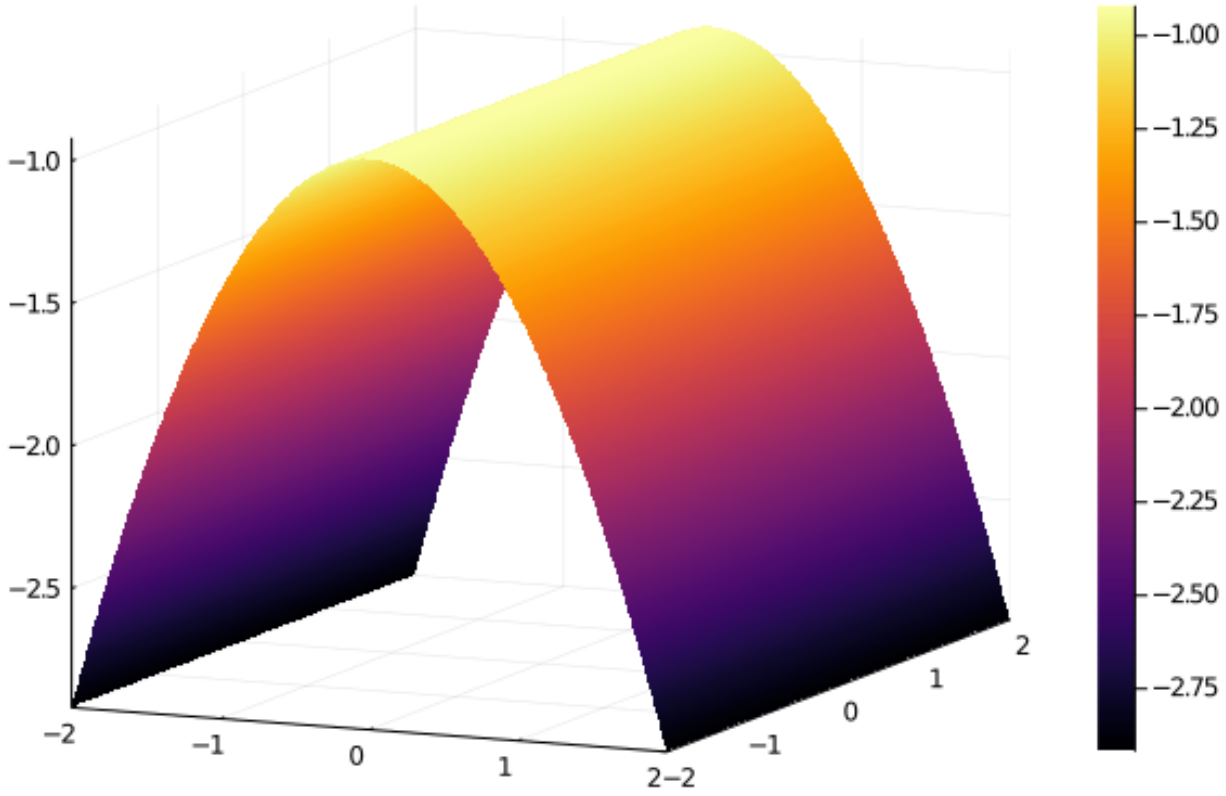
So, we can provide the MCMC sampler a better-behaved posterior geometry to explore:

$$p(y^*, x^*) = \text{Normal}(y^* \mid 0, 0) \times \prod_{n=1}^{9} \text{Normal}(x_n^* \mid 0, 0)$$
$$y = 3y^*$$
$$x_n = \exp\left(\frac{y}{2}\right) x_n^*.$$

Below there is is the Neal's Funnel reparameterized as standard normal density in 3-D.

```
kernel_reparameterized(x, y) = logpdf(Normal(), x)
surface(x, x,  kernel_reparameterized)
```

[2]this also works for multivariate distributions.



Environment

```
using InteractiveUtils
versioninfo()
using Pkg
Pkg.status()
```

```
Julia Version 1.6.0-rc1
Commit a58bdd9010* (2021-02-06 15:49 UTC)
Platform Info:
  OS: macOS (x86_64-apple-darwin20.3.0)
  CPU: Intel(R) Core(TM) i5-8500B CPU @ 3.00GHz
  WORD_SIZE: 64
  LIBM: libopenlibm
  LLVM: libLLVM-11.0.1 (ORCJIT, skylake)
Environment:
  JULIA_NUM_THREADS = 6
      Status `~/.julia/environments/v1.6/Project.toml`
  [69666777] Arrow v1.2.4
  [6e4b80f9] BenchmarkTools v0.5.0
  [336ed68f] CSV v0.8.4
  [8be319e6] Chain v0.4.4
  [a93c6f00] DataFrames v0.22.5
  [31c24e10] Distributions v0.24.15
  [366bfd00] DynamicPPL v0.10.7
  [cc61a311] FLoops v0.1.7
```

```
[c27321d9] Glob v1.3.0
[cd3eb016] HTTP v0.9.5
[b964fa9f] LaTeXStrings v1.2.1
[5078a376] LazyArrays v0.21.1
[c7f686f2] MCMCChains v4.7.0
[e1d29d7a] Missings v0.4.5
[5fb14364] OhMyREPL v0.5.10
[a15396b6] OnlineStats v1.5.8
[626c502c] Parquet v0.8.0 `https://github.com/JuliaIO/Parquet.jl#master`
[b98c9c47] Pipe v1.3.0
[ccf2f8ad] PlotThemes v2.0.1
[91a5bcdd] Plots v1.10.6
[c3e4b0f8] Pluto v0.12.21
[7f904dfe] PlutoUI v0.7.1
[37e2e3b7] ReverseDiff v1.7.0
[90137ffa] StaticArrays v1.0.1
[4c63d2b9] StatsFuns v0.9.6
[f3b207a7] StatsPlots v0.14.19
[bd369af6] Tables v1.3.2
[fce5fe82] Turing v0.15.1
[44d3d7a6] Weave v0.10.7
[e88e6eb3] Zygote v0.6.3
```

References

Neal, R. M. (2011). MCMC using Hamiltonian dynamics. In S. Brooks, A. Gelman, G. L. Jones, & X.-L. Meng (Eds.), *Handbook of markov chain monte carlo*.