# Formatting Plots

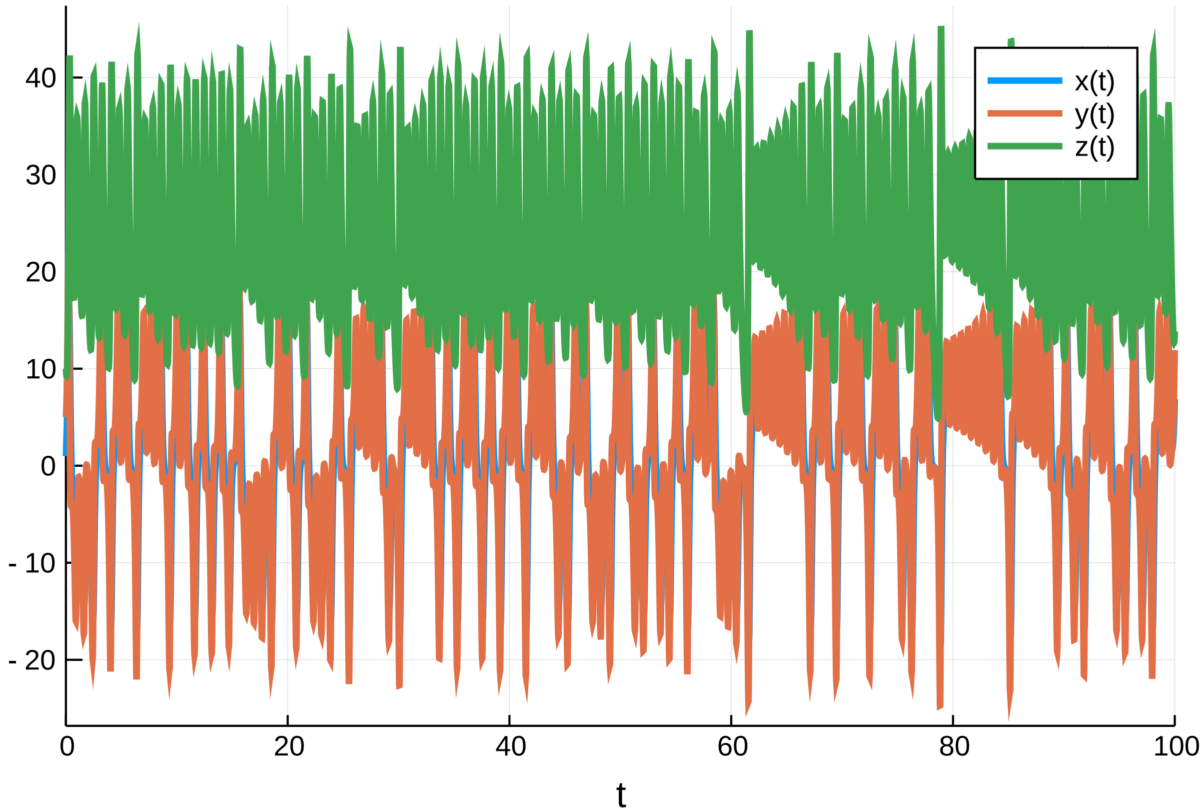
Chris Rackauckas

February 25, 2019

Since the plotting functionality is implemented as a recipe to Plots.jl, all of the options open to Plots.jl can be used in our plots. In addition, there are special features specifically for differential equation plots. This tutorial will teach some of the most commonly used options. Let's first get the solution to some ODE. Here I will use one of the Lorenz ordinary differential equation. As with all commands in DifferentialEquations.jl, I got a plot of the solution by calling `solve` on the problem, and `plot` on the solution:

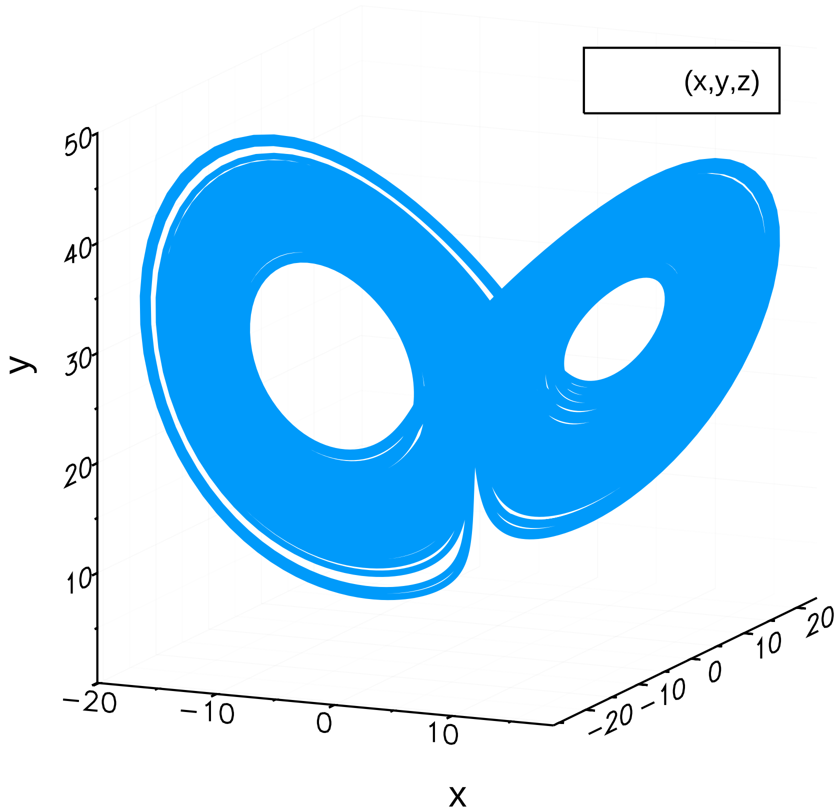
```
using DifferentialEquations, Plots, ParameterizedFunctions
gr()
lorenz = @ode_def Lorenz begin
  dx = σ*(y-x)
  dy = ρ*x-y-x*z
  dz = x*y-β*z
end σ β ρ

p = [10.0,8/3,28]
u0 = [1., 5., 10.]
tspan = (0., 100.)
prob = ODEProblem(lorenz, u0, tspan, p)
sol = solve(prob)
```

```
plot(sol)
```



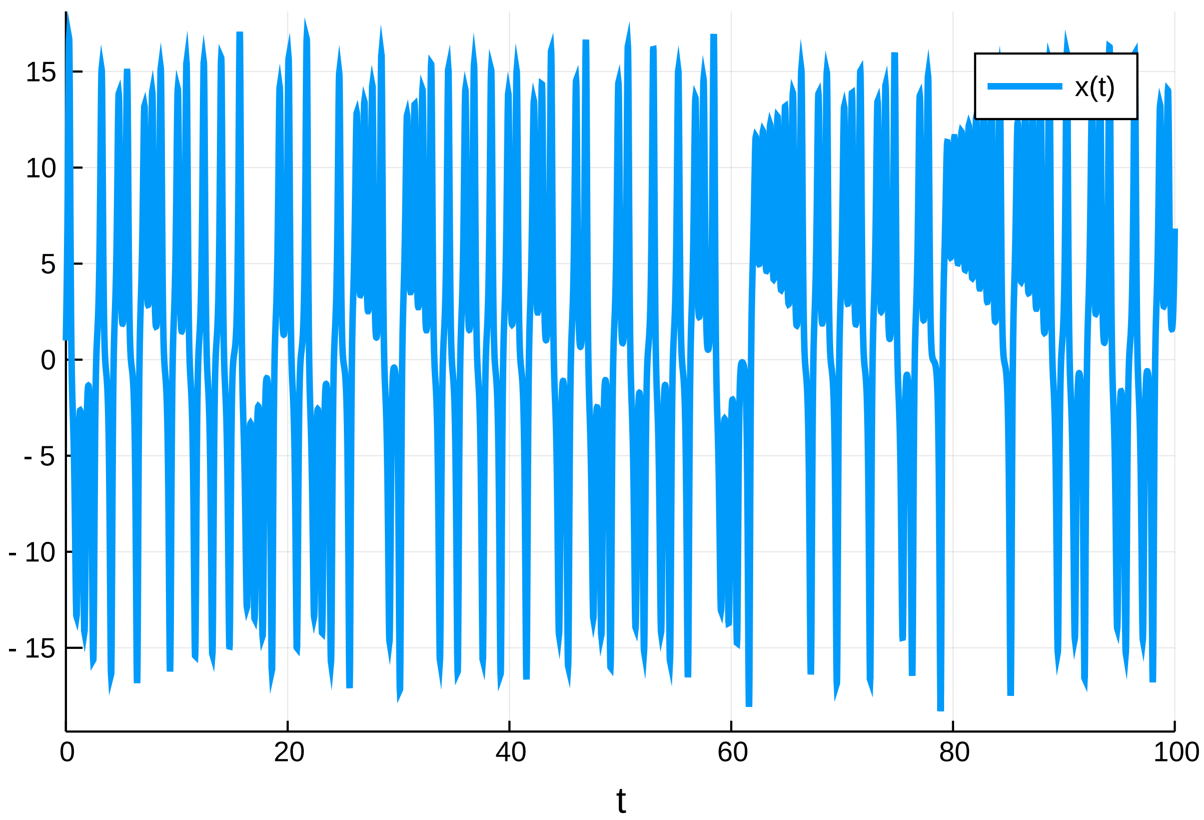

Now let's change it to a phase plot. As discussed in the plot functions page, we can use the `vars` command to choose the variables to plot. Let's plot variable `x` vs variable `y` vs variable `z`:

```
plot(sol,vars=(:x,:y,:z))
```



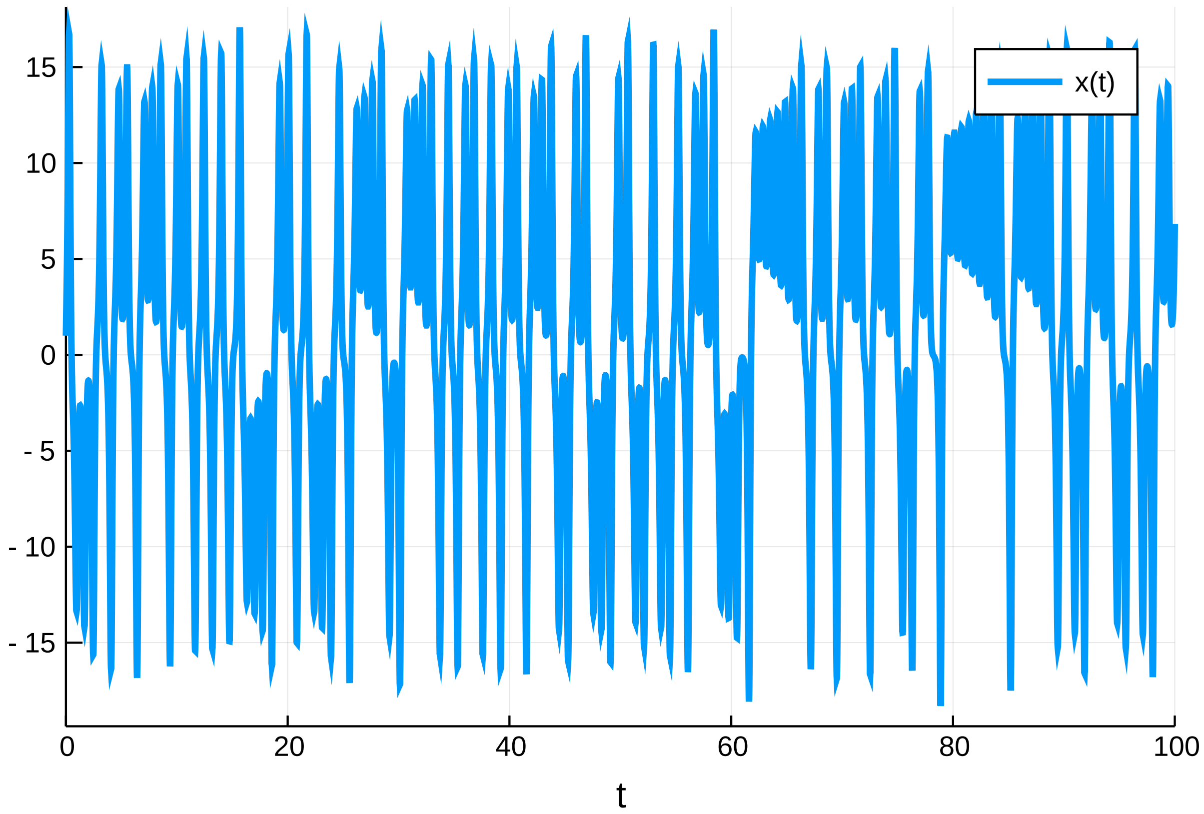

We can also choose to plot the timeseries for a single variable:

```
plot(sol,vars=[:x])
```



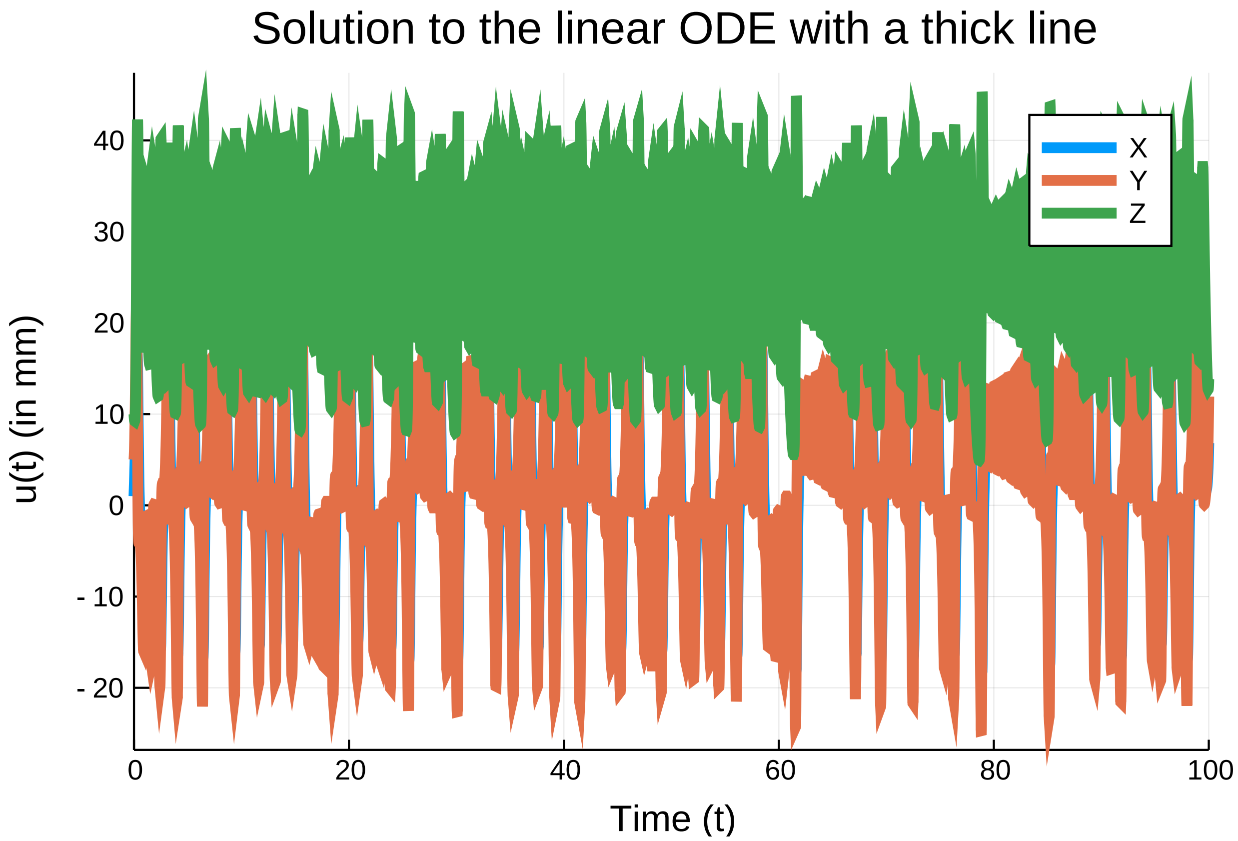
Notice that we were able to use the variable names because we had defined the problem with the macro. But in general, we can use the indices. The previous plots would be:

```
plot(sol,vars=(1,2,3))
plot(sol,vars=[1])
```



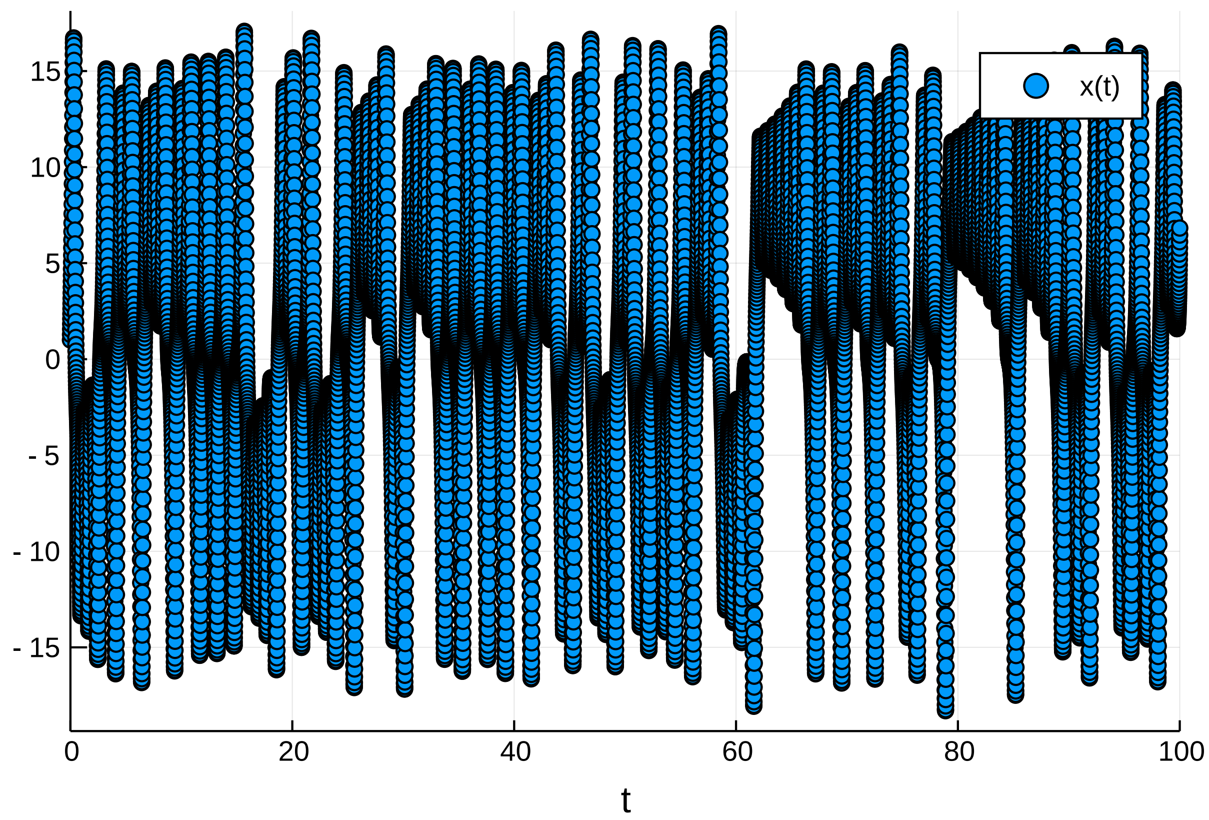

Common options are to add titles, axis, and labels. For example:

```
plot(sol,linewidth=5,title="Solution to the linear ODE with a thick line",
xaxis="Time (t)",yaxis="u(t) (in mm)",label=["X","Y","Z"])
```



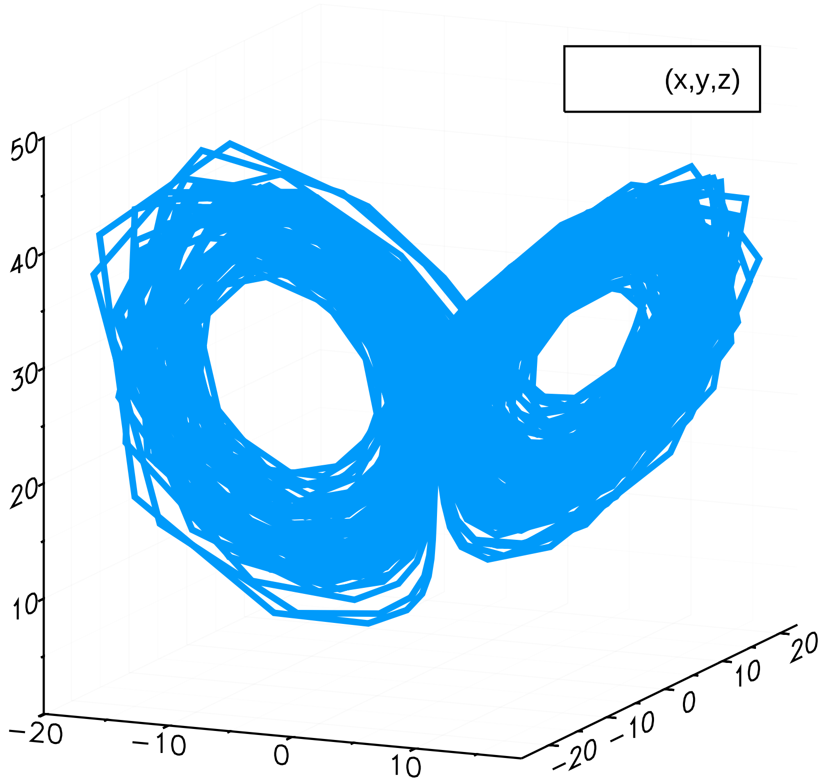

Notice that series recipes apply to the solution type as well. For example, we can use a scatter plot on the timeseries:

```
scatter(sol,vars=[:x])
```



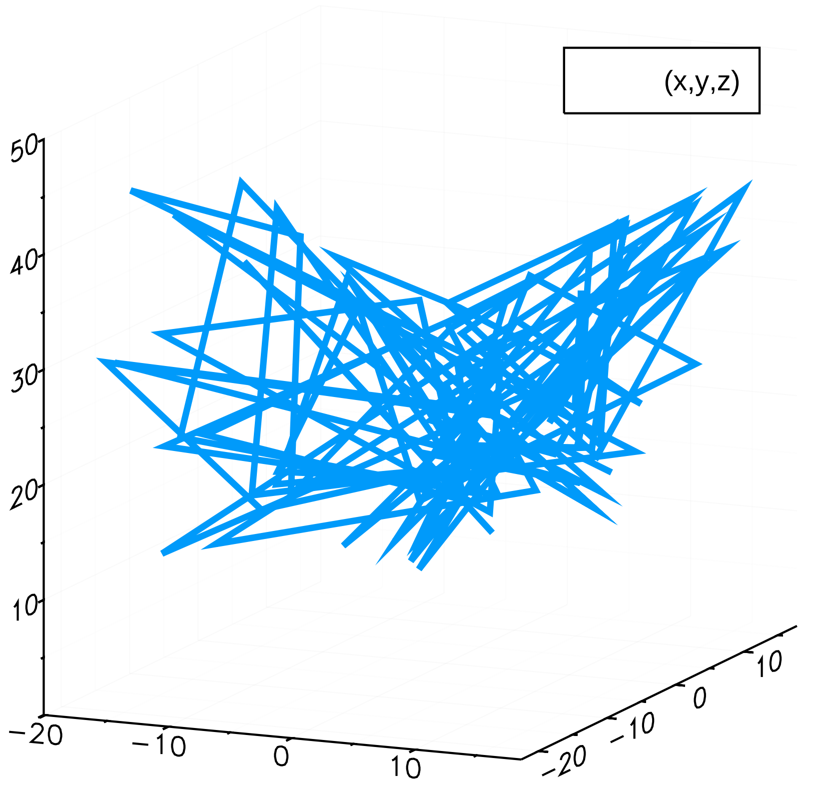
This shows that the recipe is using the interpolation to smooth the plot. It becomes abundantly clear when we turn it off using `denseplot=false`:

```
plot(sol,vars=(1,2,3),denseplot=false)
```



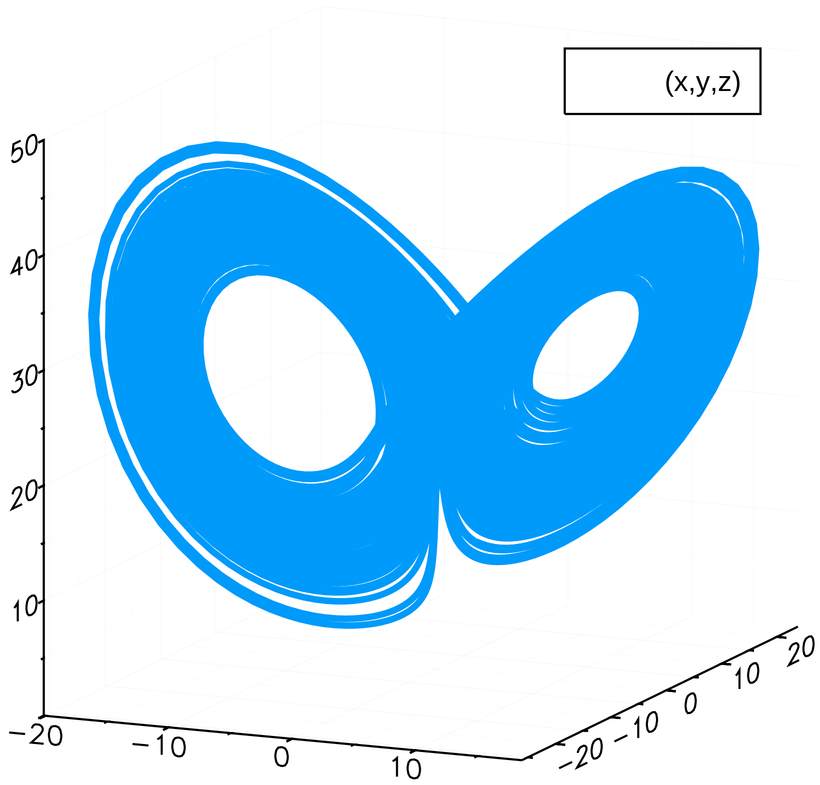

When this is done, only the values the timestep hits are plotted. Using the interpolation usually results in a much nicer looking plot so it's recommended, and since the interpolations have similar orders to the numerical methods, their results are trustworthy on the full interval. We can control the number of points used in the interpolation's plot using the `plotdensity` command:

```
plot(sol,vars=(1,2,3),plotdensity=100)
```




That's plotting the entire solution using 100 points spaced evenly in time.

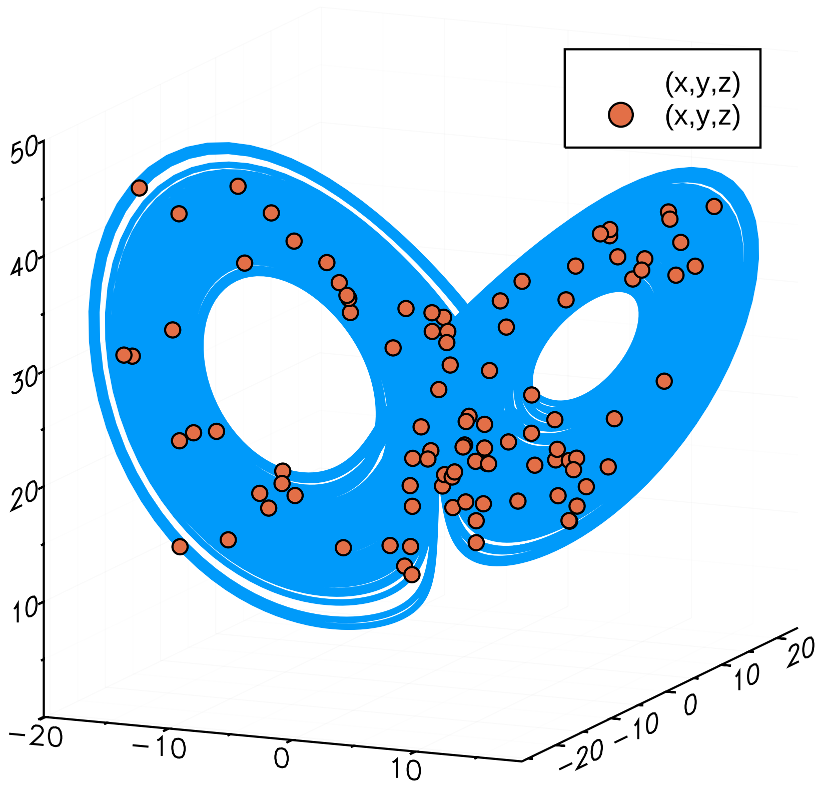
```
plot(sol,vars=(1,2,3),plotdensity=10000)
```



That's more like it! By default it uses `100*length(sol)`, where the length is the number of internal steps it had to take. This heuristic usually does well, but unusually difficult equations it can be relaxed (since it will take small steps), and for equations with events / discontinuities raising the plot density can help resolve the discontinuity.

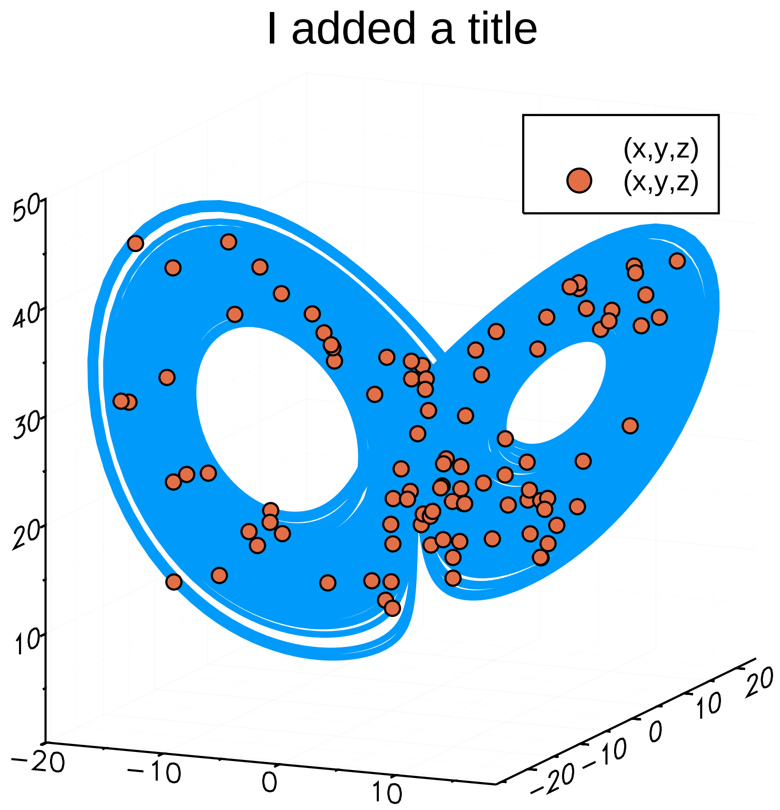
Lastly notice that we can compose plots. Let's show where the 100 points are using a scatter plot:

```
plot(sol,vars=(1,2,3))
scatter!(sol,vars=(1,2,3),plotdensity=100)
```




We can instead work with an explicit plot object. This form can be better for building a complex plot in a loop.

```
p = plot(sol,vars=(1,2,3))
scatter!(p,sol,vars=(1,2,3),plotdensity=100)
title!("I added a title")
```




You can do all sorts of things. Have fun!

## 0.1 Appendix

```
using DiffEqTutorials
DiffEqTutorials.tutorial_footer(WEAVE_ARGS[:folder],WEAVE_ARGS[:file])
```

```
These benchmarks are part of the DiffEqTutorials.jl repository, found at:

https://github.com/JuliaDiffEq/DiffEqTutorials.jl

To locally run this tutorial, do the following commands:

using DiffEqTutorials
DiffEqTutorials.weave_file("introduction","formatting_plots.jmd")

Computer Information:

Julia Version 1.1.0
Commit 80516ca202 (2019-01-21 21:24 UTC)
Platform Info:
  OS: Windows (x86_64-w64-mingw32)
  CPU: Intel(R) Core(TM) i7-8700 CPU @ 3.20GHz
  WORD_SIZE: 64
  LIBM: libopenlibm
  LLVM: libLLVM-6.0.1 (ORCJIT, skylake)
Environment:
  JULIA_EDITOR = "C:\Users\accou\AppData\Local\atom\app-1.34.0\atom.exe" -a
  JULIA_NUM_THREADS = 6
```

Package Information:

```
  Status `C:\Users\accou\.julia\environments\v1.1\Project.toml`
  [7e558dbc] ArbNumerics v0.3.6
  [c52e3926] Atom v0.7.14
  [6e4b80f9] BenchmarkTools v0.4.2
  [336ed68f] CSV v0.4.3
  [3895d2a7] CUDAapi v0.5.4
  [be33ccc6] CUDAnative v1.0.1
  [3a865a2d] CuArrays v0.9.1
  [a93c6f00] DataFrames v0.17.1
  [55939f99] DecFP v0.4.8
  [abce61dc] Decimals v0.4.0
  [39dd38d3] Dierckx v0.4.1
  [459566f4] DiffEqCallbacks v2.5.2
  [f3b72e0c] DiffEqDevTools v2.6.1
  [aae7a2af] DiffEqFlux v0.2.0
  [c894b116] DiffEqJump v6.1.0+ [`C:\Users\accou\.julia\dev\DiffEqJump`]
  [1130ab10] DiffEqParamEstim v1.6.0+ [`C:\Users\accou\.julia\dev\DiffEqPar
amEstim`]
  [055956cb] DiffEqPhysics v3.1.0
  [225cb15b] DiffEqTutorials v0.0.0 [`C:\Users\accou\.julia\external\DiffEq
Tutorials.jl`]
  [0c46a032] DifferentialEquations v6.3.0
  [497a8b3b] DoubleFloats v0.7.5
  [587475ba] Flux v0.7.3
  [f6369f11] ForwardDiff v0.10.3+ [`C:\Users\accou\.julia\dev\ForwardDiff`]
  [28b8d3ca] GR v0.38.1
  [7073ff75] IJulia v1.17.0
  [c601a237] Interact v0.9.1
  [b6b21f68] Ipopt v0.5.4
  [4076af6c] JuMP v0.19.0
  [e5e0dc1b] Juno v0.5.4
  [eff96d63] Measurements v2.0.0
  [76087f3c] NLopt v0.5.1
  [429524aa] Optim v0.17.2
  [1dea7af3] OrdinaryDiffEq v5.2.1+ [`C:\Users\accou\.julia\dev\OrdinaryDif
fEq`]
  [65888b18] ParameterizedFunctions v4.1.1
  [91a5bcdd] Plots v0.23.0
  [71ad9d73] PuMaS v0.0.0 [`C:\Users\accou\.julia\dev\PuMaS`]
  [d330b81b] PyPlot v2.7.0
  [731186ca] RecursiveArrayTools v0.20.0
  [90137ffa] StaticArrays v0.10.2
  [789caeaf] StochasticDiffEq v6.1.1+ [`C:\Users\accou\.julia\dev\Stochasti
cDiffEq`]
  [c3572dad] Sundials v3.0.0
  [1986cc42] Unitful v0.14.0
  [2a06ce6d] UnitfulPlots v0.0.0 #master (https://github.com/ajkeller34/Uni
tfulPlots.jl)
  [44d3d7a6] Weave v0.7.1 [`C:\Users\accou\.julia\dev\Weave`]
```