Pumas NCA Tutorial - Analyzing infusion data

Beatriz Guglieri Lopez, Shamir Kalaria, Vijay Ivaturi

July 19, 2019

```
using Pumas, PumasTutorials, CSV, Plots
```

1 Introduction

In this tutorial, we will cover the fundamentals of performing an NCA analysis with Pumas using an example dataset in which an intravenous infusion was administered.

2 The dataset

- Single 2000 mg 2-hour IV infusion dose to 24 subjects
- Blood samples for pharmacokinetic analysis were collected every 30 minutes

```
data = PumasTutorials.tutorial_data("data/nca","SD_IV_infusion")
data = CSV.read(data,missingstring="NA")
first(data,10)
```

	ID	time	DV	BLQ	DOSE	Infusion_Time	Formulation
	Int64	Float64	Float64	Int64	Int64	Int64	String
1	1	0.0	0.0	0	2000	2	iv
2	1	0.5	9.1425	0	0	0	iv
3	1	1.0	17.9045	0	0	0	iv
4	1	1.5	25.2709	0	0	0	iv
5	1	2.0	29.4367	0	0	0	iv
6	1	2.5	29.1849	0	0	0	iv
7	1	3.0	26.8687	0	0	0	iv
8	1	3.5	20.8217	0	0	0	iv
9	1	4.0	19.6429	0	0	0	iv
10	1	4.5	17.5575	0	0	0	iv

Please, note that the `DOSE` and the `Infusion_Time` column must only contain a value at the time the dose was administered.

3 Defining the units

```
timeu = u"hr"
concu = u"mg/L"
amtu  = u"mg"
```

```
mg
```

4 Defining the population object

In the case of the infusion, the `read_nca` function should

- carry the name of the column that contains the duration of the infusion (`duration=`).

```
pop = read_nca(data, id=:ID, time=:time, conc=:DV, amt=:DOSE, ii=24timeu,
    route=:Formulation, duration=:Infusion_Time,timeu=timeu,
    concu=concu, amtu=amtu,llq=0.4concu)
```

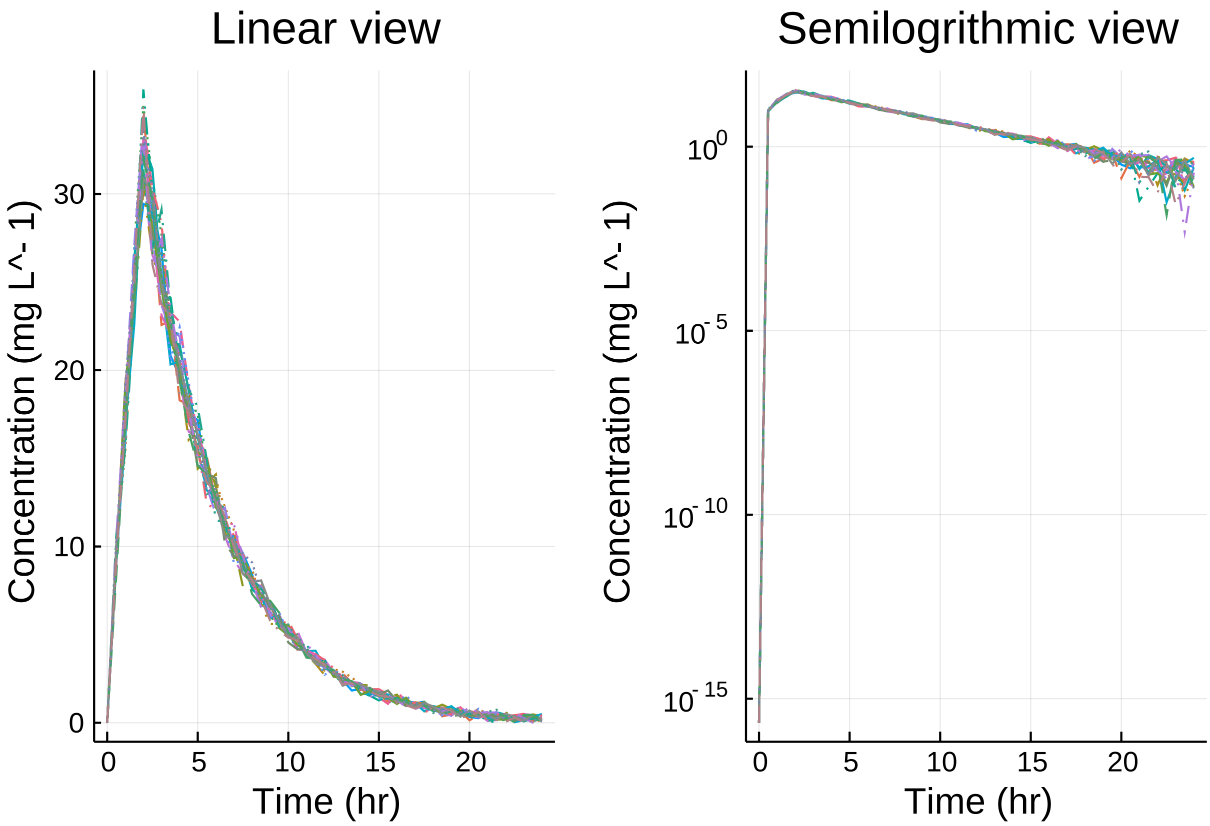
```
NCAPopulation (24 subjects):
  ID: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 2
0, 21, 22, 23, 24]
    concentration: mg L^-1
    time:          hr
    auc:           mg hr L^-1
    aumc:          mg hr^2 L^-1
    λz:            hr^-1
    dose:          mg
```

Note that in the above syntanx:

- `route=` is mapped to the `Formulation` column that should specify `iv`
- LLOQ was set to 0.4 by `llq=0.4concu`

A basic plot function exists for single dose data without grouping or multiple analytes. More functionality will be added soon. In this example of single dose data, here is the plot output

```
plot(pop)
```



5 NCA functions

5.0.1 NCA.auc

```
NCA.auc(pop,auctype=:last,method=:linear)
```

	id	auc
	Int64	Unitful
1	1	170.557 mg hr L^-1
2	2	171.183 mg hr L^-1
3	3	177.307 mg hr L^-1
4	4	170.187 mg hr L^-1
5	5	168.923 mg hr L^-1
6	6	174.448 mg hr L^-1
7	7	177.059 mg hr L^-1
8	8	173.479 mg hr L^-1
9	9	174.841 mg hr L^-1
10	10	170.972 mg hr L^-1
11	11	168.468 mg hr L^-1
12	12	173.454 mg hr L^-1
13	13	172.529 mg hr L^-1
14	14	174.262 mg hr L^-1
15	15	172.386 mg hr L^-1
16	16	169.952 mg hr L^-1
17	17	170.451 mg hr L^-1
18	18	173.565 mg hr L^-1
19	19	174.075 mg hr L^-1
20	20	174.062 mg hr L^-1
21	21	172.185 mg hr L^-1
22	22	169.771 mg hr L^-1
23	23	176.241 mg hr L^-1
24	24	168.571 mg hr L^-1

To change the methods to log-linear trapezoidal (`method=:linuplogdown`) or to linear-log (`method=:linlog`) one can use

```
NCA.auc(pop,auctype=:inf,method=:linuplogdown)
```

To compute the AUC over an interval, one could do

```
NCA.auc(pop, interval=(0,12).*timeu)
```

where we need to apply the time unit (`timeu`) to the `interval` for units compatibility. Multiple intervals can also be specified:

```
NCA.auc(pop, interval=[(0,12).*timeu,(0,6).*timeu])
```

5.0.2 `NCA.lambdaz`

The function to calculate the terminal rate constant (λz) is:

```
NCA.lambdaz(pop)
```

This function has options that allow

- to specify the maximum number of points to be used for lambdaz - `threshold=3`
- calculation to be peformed over specified indices - `idxs=[18,19,20]` where index 18,19,20 of the subject will be used for lambdaz
- speification of exact time points to use for lambdaz - `slopetimes=[18.5,19,19.5].*timeu`

```
NCA.lambdaz(pop, threshold=3)
NCA.lambdaz(pop, idxs=[18,19,20])
NCA.lambdaz(pop, slopetimes=[18.5,19,19.5].*timeu)
```

5.0.3 NCA.cmax

To calculate the maximum concentration for the first subject we would use:

```
cmax = NCA.cmax(pop[1])
```

```
29.4367368 mg L^-1
```

5.0.4 NCA.normalizedose

If we want dose-normalized Cmax for that same subject:

```
NCA.normalizedose(cmax,pop[1])
```

```
0.0147183684 L^-1
```

This can be used on any parameter that can be dose normalized.

Other functions to calculate single PK parameters are the following:

```
NCA.lambdazr2(pop)
NCA.lambdazadjr2(pop)
NCA.lambdazintercept(pop)
NCA.lambdaztimefirst(pop)
NCA.lambdaznpoints(pop)

NCA.tmax(pop)
NCA.cmin(pop)
NCA.tmin(pop)

NCA.tlast(pop)
NCA.clast(pop)

NCA.aumc(pop)
NCA.aumclast(pop)

NCA.thalf(pop)

NCA.cl(pop)

NCA.vss(pop)
NCA.vz(pop)
```

	id	vz
	Int64	Unitful
1	1	30.6639 L
2	2	32.3386 L
3	3	10.3627 L
4	4	24.7198 L
5	5	-19.3688 L
6	6	37.0702 L
7	7	45.5432 L
8	8	22.8918 L
9	9	-23.0548 L
10	10	33.1127 L
11	11	-9.75512 L
12	12	32.7396 L
13	13	46.5234 L
14	14	54.7095 L
15	15	-34.0404 L
16	16	7.99809 L
17	17	26.6045 L
18	18	-27.3057 L
19	19	39.0445 L
20	20	16.3506 L
21	21	43.5989 L
22	22	6.49356 L
23	23	8.78626 L
24	24	-4.95184 L

6 NCA report

If we want a complete report of the NCA analysis we can just use the function `NCAreport` to obtain a data frame that contains all the above mentioned pharmacokinetic parameters.

```
report = NCAReport(pop)
report = NCA.to_dataframe(report)
```

	id	doseamt	lambda_z	half_life	tmax	cmax	clast
	Int64	Unitful	Unitful	Unitful	Unitful	Unitful	Unitful
1	1	2000 mg	0.380768 hr^-1	1.82039 hr	2.0 hr	29.4367 mg L^-1	0.280525 mg L^-1
2	2	2000 mg	0.359903 hr^-1	1.92593 hr	2.0 hr	34.6225 mg L^-1	0.23629 mg L^-1
3	3	2000 mg	1.08775 hr^-1	0.637233 hr	2.0 hr	34.9916 mg L^-1	0.13635 mg L^-1
4	4	2000 mg	0.473609 hr^-1	1.46354 hr	2.0 hr	31.1252 mg L^-1	0.304896 mg L^-1
5	5	2000 mg	-0.613372 hr^-1	-1.13006 hr	2.0 hr	30.2427 mg L^-1	0.354381 mg L^-1
6	6	2000 mg	0.307699 hr^-1	2.25268 hr	2.0 hr	32.8505 mg L^-1	0.274121 mg L^-1
7	7	2000 mg	0.246005 hr^-1	2.81761 hr	2.0 hr	34.0576 mg L^-1	0.35681 mg L^-1
8	8	2000 mg	0.502704 hr^-1	1.37884 hr	2.0 hr	33.253 mg L^-1	0.159291 mg L^-1
9	9	2000 mg	-0.497009 hr^-1	-1.39464 hr	2.0 hr	35.928 mg L^-1	0.147872 mg L^-1
10	10	2000 mg	0.351833 hr^-1	1.9701 hr	2.0 hr	30.314 mg L^-1	0.246073 mg L^-1
11	11	2000 mg	-1.21993 hr^-1	-0.568187 hr	2.0 hr	32.7331 mg L^-1	0.498754 mg L^-1
12	12	2000 mg	0.350243 hr^-1	1.97905 hr	2.0 hr	34.1103 mg L^-1	0.337286 mg L^-1
13	13	2000 mg	0.247839 hr^-1	2.79676 hr	2.0 hr	30.7831 mg L^-1	0.229543 mg L^-1
14	14	2000 mg	0.207269 hr^-1	3.3442 hr	2.0 hr	30.4458 mg L^-1	0.437615 mg L^-1
15	15	2000 mg	-0.342741 hr^-1	-2.02237 hr	2.0 hr	33.1621 mg L^-1	0.329866 mg L^-1
16	16	2000 mg	1.47088 hr^-1	0.471246 hr	2.0 hr	30.9219 mg L^-1	0.080925 mg L^-1
17	17	2000 mg	0.439428 hr^-1	1.57739 hr	2.0 hr	31.4966 mg L^-1	0.274488 mg L^-1
18	18	2000 mg	-0.424339 hr^-1	-1.63348 hr	2.0 hr	33.1599 mg L^-1	0.405454 mg L^-1
19	19	2000 mg	0.29365 hr^-1	2.36046 hr	2.0 hr	31.3835 mg L^-1	0.106554 mg L^-1
20	20	2000 mg	0.701601 hr^-1	0.987951 hr	2.0 hr	32.7522 mg L^-1	0.197867 mg L^-1
21	21	2000 mg	0.265666 hr^-1	2.6091 hr	2.0 hr	30.867 mg L^-1	0.129004 mg L^-1
22	22	2000 mg	1.81378 hr^-1	0.382157 hr	2.0 hr	31.9961 mg L^-1	0.0710516 mg L^-1
23	23	2000 mg	1.29115 hr^-1	0.536844 hr	2.0 hr	33.5559 mg L^-1	0.0743402 mg L^-1
24	24	2000 mg	-2.39769 hr^-1	-0.289089 hr	2.0 hr	35.0072 mg L^-1	0.291334 mg L^-1

By default, the AUC and AUMC reported are observed. If predicted PK parameters are needed instead, the following code should be used:

```
report = NCAReport(pop,pred=true)
report = NCA.to_dataframe(report)
```

	id	doseamt	lambda_z	half_life	tmax	cmax	clast
	Int64	Unitful	Unitful	Unitful	Unitful	Unitful	Unitful
1	1	2000 mg	0.380768 hr^-1	1.82039 hr	2.0 hr	29.4367 mg L^-1	0.280525 mg L^-1
2	2	2000 mg	0.359903 hr^-1	1.92593 hr	2.0 hr	34.6225 mg L^-1	0.23629 mg L^-1
3	3	2000 mg	1.08775 hr^-1	0.637233 hr	2.0 hr	34.9916 mg L^-1	0.13635 mg L^-1
4	4	2000 mg	0.473609 hr^-1	1.46354 hr	2.0 hr	31.1252 mg L^-1	0.304896 mg L^-1
5	5	2000 mg	-0.613372 hr^-1	-1.13006 hr	2.0 hr	30.2427 mg L^-1	0.354381 mg L^-1
6	6	2000 mg	0.307699 hr^-1	2.25268 hr	2.0 hr	32.8505 mg L^-1	0.274121 mg L^-1
7	7	2000 mg	0.246005 hr^-1	2.81761 hr	2.0 hr	34.0576 mg L^-1	0.35681 mg L^-1
8	8	2000 mg	0.502704 hr^-1	1.37884 hr	2.0 hr	33.253 mg L^-1	0.159291 mg L^-1
9	9	2000 mg	-0.497009 hr^-1	-1.39464 hr	2.0 hr	35.928 mg L^-1	0.147872 mg L^-1
10	10	2000 mg	0.351833 hr^-1	1.9701 hr	2.0 hr	30.314 mg L^-1	0.246073 mg L^-1
11	11	2000 mg	-1.21993 hr^-1	-0.568187 hr	2.0 hr	32.7331 mg L^-1	0.498754 mg L^-1
12	12	2000 mg	0.350243 hr^-1	1.97905 hr	2.0 hr	34.1103 mg L^-1	0.337286 mg L^-1
13	13	2000 mg	0.247839 hr^-1	2.79676 hr	2.0 hr	30.7831 mg L^-1	0.229543 mg L^-1
14	14	2000 mg	0.207269 hr^-1	3.3442 hr	2.0 hr	30.4458 mg L^-1	0.437615 mg L^-1
15	15	2000 mg	-0.342741 hr^-1	-2.02237 hr	2.0 hr	33.1621 mg L^-1	0.329866 mg L^-1
16	16	2000 mg	1.47088 hr^-1	0.471246 hr	2.0 hr	30.9219 mg L^-1	0.080925 mg L^-1
17	17	2000 mg	0.439428 hr^-1	1.57739 hr	2.0 hr	31.4966 mg L^-1	0.274488 mg L^-1
18	18	2000 mg	-0.424339 hr^-1	-1.63348 hr	2.0 hr	33.1599 mg L^-1	0.405454 mg L^-1
19	19	2000 mg	0.29365 hr^-1	2.36046 hr	2.0 hr	31.3835 mg L^-1	0.106554 mg L^-1
20	20	2000 mg	0.701601 hr^-1	0.987951 hr	2.0 hr	32.7522 mg L^-1	0.197867 mg L^-1
21	21	2000 mg	0.265666 hr^-1	2.6091 hr	2.0 hr	30.867 mg L^-1	0.129004 mg L^-1
22	22	2000 mg	1.81378 hr^-1	0.382157 hr	2.0 hr	31.9961 mg L^-1	0.0710516 mg L^-1
23	23	2000 mg	1.29115 hr^-1	0.536844 hr	2.0 hr	33.5559 mg L^-1	0.0743402 mg L^-1
24	24	2000 mg	-2.39769 hr^-1	-0.289089 hr	2.0 hr	35.0072 mg L^-1	0.291334 mg L^-1

Finally, we can save this data frame as a csv file if desired.

```
CSV.write("./tutorials/nca/report_SD_IV_infusion.csv", report)
```

```
Error: SystemError: opening file "./tutorials/nca/report_SD_IV_infusion.csv
": No such file or directory
```

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

6.1 Appendix

These tutorials are part of the PumasTutorials.jl repository, found at: https://github.com/JuliaDiffEq/Di

To locally run this tutorial, do the following commands:

```
using PumasTutorials
PumasTutorials.weave_file("nca","SD_IV_infusion.jmd")
```

Computer Information:

```
Julia Version 1.1.1
Commit 55e36cc308 (2019-05-16 04:10 UTC)
```

```
Platform Info:
  OS: Windows (x86_64-w64-mingw32)
  CPU: Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz
  WORD_SIZE: 64
  LIBM: libopenlibm
  LLVM: libLLVM-6.0.1 (ORCJIT, skylake)
Environment:
  JULIA_EDITOR = "C:\Users\accou\AppData\Local\atom\app-1.38.2\atom.exe"  -a
  JULIA_NUM_THREADS = 4
```

Package Information:

```
Status `C:\Users\accou\.julia\environments\v1.1\Project.toml`
[621f4979-c628-5d54-868e-fcf4e3e8185c] AbstractFFTs 0.4.1
[c52e3926-4ff0-5f6e-af25-54175e0327b1] Atom 0.8.8
[f0abef60-9ec0-11e9-27de-db6506a91768] AutoOffload 0.1.0
[6e4b80f9-dd63-53aa-95a3-0cdb28fa8baf] BenchmarkTools 0.4.2
[4ece37e6-a012-11e8-38cd-91247efc2c34] Bioequivalence 0.1.0
[336ed68f-0bac-5ca0-87d4-7b16caf5d00b] CSV 0.5.9
[c5f51814-7f29-56b8-a69c-e4d8f6be1fde] CUDAdrv 3.0.1
[be33ccc6-a3ff-5ff2-a52e-74243cff1e17] CUDAnative 2.2.1
[49dc2e85-a5d0-5ad3-a950-438e2897f1b9] Calculus 0.5.0
[7057c7e9-c182-5462-911a-8362d720325c] Cassette 0.2.5
[34da2185-b29b-5c13-b0c7-acf172513d20] Compat 2.1.0
[3a865a2d-5b23-5a0f-bc46-62713ec82fae] CuArrays 1.1.0
[667455a9-e2ce-5579-9412-b964f529a492] Cubature 1.4.0
[a93c6f00-e57d-5684-b7b6-d8193f3e46c0] DataFrames 0.18.4
[82cc6244-b520-54b8-b5a6-8a565e85f1d0] DataInterpolations 0.2.0
[31a5f54b-26ea-5ae9-a837-f05ce5417438] Debugger 0.5.0
[bcd4f6db-9728-5f36-b5f7-82caef46ccdb] DelayDiffEq 5.9.1
[2b5f629d-d688-5b77-993f-72d75c75574e] DiffEqBase 5.16.3
[ebbdde9d-f333-5424-9be2-dbf1e9acfb5e] DiffEqBayes 1.2.0
[31c91b34-3c75-11e9-0341-95557aab0344] DiffEqBenchmarks 0.1.0
[459566f4-90b8-5000-8ac3-15dfb0a30def] DiffEqCallbacks 2.5.2+
[f3b72e0c-5b89-59e1-b016-84e28bfd966d] DiffEqDevTools 2.13.0
[01453d9d-ee7c-5054-8395-0335cb756afa] DiffEqDiffTools 0.14.0
[aae7a2af-3d4f-5e19-a356-7da93b79d9d0] DiffEqFlux 0.6.0
[071ae1c0-96b5-11e9-1965-c90190d839ea] DiffEqGPU 0.1.0
[c894b116-72e5-5b58-be3c-e6d8d4ac2b12] DiffEqJump 6.1.1+
[8f2b45d5-b17b-5532-9e92-98ae0077e2e3] DiffEqMachineLearning 0.1.0
[78ddff82-25fc-5f2b-89aa-309469cbf16f] DiffEqMonteCarlo 0.15.1
[77a26b50-5914-5dd7-bc55-306e6241c503] DiffEqNoiseProcess 3.3.1
[9fdde737-9c7f-55bf-ade8-46b3f136cc48] DiffEqOperators 3.5.0
[055956cb-9e8b-5191-98cc-73ae4a59e68a] DiffEqPhysics 3.2.0
[a077e3f3-b75c-5d7f-a0c6-6bc4c8ec64a9] DiffEqProblemLibrary 4.3.0
[41bf760c-e81c-5289-8e54-58b1f1f8abe2] DiffEqSensitivity 3.3.0
[6d1b261a-3be8-11e9-3f2f-0b112a9a8436] DiffEqTutorials 0.1.0
[0c46a032-eb83-5123-abaf-570d42b7fbaa] DifferentialEquations 6.6.0
```

```
[31c24e10-a181-5473-b8eb-7969acd0382f] Distributions 0.20.0
[e30172f5-a6a5-5a46-863b-614d45cd2de4] Documenter 0.23.0
[587475ba-b771-5e3f-ad9e-33799f191a9c] Flux 0.8.3
[f6369f11-7733-5829-9624-2563aa707210] ForwardDiff 0.10.3+
[ba82f77b-6841-5d2e-bd9f-4daf811aec27] GPUifyLoops 0.2.5
[c91e804a-d5a3-530f-b6f0-dfbca275c004] Gadfly 1.1.0
[bc5e4493-9b4d-5f90-b8aa-2b2bcaad7a26] GitHub 5.1.1
[7073ff75-c697-5162-941a-fcdaad2a7d2a] IJulia 1.18.1
[42fd0dbc-a981-5370-80f2-aaf504508153] IterativeSolvers 0.8.1
[033835bb-8acc-5ee8-8aae-3f567f8a3819] JLD2 0.1.2
[e5e0dc1b-0480-54bc-9374-aad01c23163d] Juno 0.7.0
[2d691ee1-e668-5016-a719-b2531b85e0f5] LIBLINEAR 0.5.1
[7f56f5a3-f504-529b-bc02-0b1fe5e64312] LSODA 0.4.0
[6f1fad26-d15e-5dc8-ae53-837a1d7b8c9f] Libtask 0.3.0
[c7f686f2-ff18-58e9-bc7b-31028e88f75d] MCMCChains 0.3.10
[33e6dc65-8f57-5167-99aa-e5a354878fb2] MKL 0.0.0
[cc2ba9b6-d476-5e6d-8eaf-a92d5412d41d] MLDataUtils 0.5.0
[eb30cadb-4394-5ae3-aed4-317e484a6458] MLDatasets 0.3.0
[961ee093-0014-501f-94e3-6117800e7a78] ModelingToolkit 0.5.0
[4886b29c-78c9-11e9-0a6e-41e1f4161f7b] MonteCarloIntegration 0.0.1
[2774e3e8-f4cf-5e23-947b-6d7e65073b56] NLsolve 4.0.0
[872c559c-99b0-510c-b3b7-b6c96a88d5cd] NNlib 0.6.0
[8faf48c0-8b73-11e9-0e63-2155955bfa4d] NeuralNetDiffEq 0.1.0
[09606e27-ecf5-54fc-bb29-004bd9f985bf] ODEInterfaceDiffEq 3.3.1
[1dea7af3-3e70-54e6-95c3-0bf5283fa5ed] OrdinaryDiffEq 5.12.0
[65888b18-ceab-5e60-b2b9-181511a3b968] ParameterizedFunctions 4.2.0
[14b8a8f1-9102-5b29-a752-f990bacb7fe1] PkgTemplates 0.6.1
[91a5bcdd-55d7-5caf-9e0b-520d859cae80] Plots 0.25.3
[92933f4c-e287-5a05-a399-4b506db050ca] ProgressMeter 1.0.0
[d7b8c89e-ad89-52e0-b9fd-d0ed321fa021] Pumas 0.1.0
[b7b41870-aa11-11e9-048a-09266ec4a62f] PumasTutorials 0.0.1
[438e738f-606a-5dbb-bf0a-cddfbfd45ab0] PyCall 1.91.2
[d330b81b-6aea-500a-939a-2ce795aea3ee] PyPlot 2.8.1
[1fd47b50-473d-5c70-9696-f719f8f3bcdc] QuadGK 2.1.0
[612083be-0b0f-5412-89c1-4e7c75506a58] Queryverse 0.3.1
[6f49c342-dc21-5d91-9882-a32aef131414] RCall 0.13.3
[731186ca-8d62-57ce-b412-fbd966d074cd] RecursiveArrayTools 0.20.0
[37e2e3b7-166d-5795-8a7a-e32c996b4267] ReverseDiff 0.3.1
[295af30f-e4ad-537b-8983-00126c2a3abe] Revise 2.1.6
[2b6d1eac-7baa-5078-8adc-e6a3e659f14f] SingleFloats 0.1.3
[47a9eef4-7e08-11e9-0b38-333d64bd3804] SparseDiffTools 0.5.0
[90137ffa-7385-5640-81b9-e52037218182] StaticArrays 0.11.0
[4c63d2b9-4356-54db-8cca-17b64c39e42c] StatsFuns 0.8.0
[f3b207a7-027a-5e70-b257-86293d7955fd] StatsPlots 0.11.0
[9672c7b4-1e72-59bd-8a11-6ac3964bc41f] SteadyStateDiffEq 1.5.0
[789caeaf-c7a9-5a7d-9973-96adeb23e2a0] StochasticDiffEq 6.6.0
[c3572dad-4567-51f8-b174-8c6c989267f4] Sundials 3.6.1
[fd094767-a336-5f1f-9728-57cf17d0bbfb] Suppressor 0.1.1
[6fc51010-71bc-11e9-0e15-a3fcc6593c49] Surrogates 0.1.0
```

```
[9f7883ad-71c0-57eb-9f7f-b5c9e6d3789c] Tracker 0.2.2
[fce5fe82-541a-59a6-adf8-730c64b5f9a0] Turing 0.6.18
[1986cc42-f94f-5a68-af5c-568840ba703d] Unitful 0.16.0
[44d3d7a6-8a23-5bf8-98c5-b353f8df5ec9] Weave 0.9.1
[e88e6eb3-aa80-5325-afca-941959d7151f] Zygote 0.3.2
```