A Complete Bayesian Analysis Workflow in Pumas

Mohamed Tarek, Ph.D.

Senior Product Engineer, Pumas-AI

mohamed@pumas.ai

PAGANZ 2023

pumas AI Healthcare Intelligence Augmented

Tutorial Paper Coming Soon!

A Practitioner's Guide to Bayesian Inference in Pharmacometrics using Pumas

Mohamed Tarek[1,2*], Jose Storopoli[1,3], Casey Davis[4], Chris Elrod[1,5], Julius Krumbiegel[1] and Vijay Ivaturi[1,6]

[1*]Pumas-AI Inc., USA.
[2]Business School, University of Sydney, Australia.
[3]Universidade Nove de Julho - UNINOVE, Brazil.
[4]Gilead Sciences Inc., USA.
[5]JuliaHub (formerly JuliaComputing) Inc., USA.
[6]University of Maryland Baltimore, USA.

*Corresponding author(s). E-mail(s): mohamed@pumas.ai;
Contributing authors: jose@pumas.ai; cbdavis33@gmail.com; chrise@pumas.ai; juliusk@pumas.ai; vijay@pumas.ai;

Most scientists are concerned with inefficient workflows current software offer, so we created a powerful integrated software which double your think time

pumasAI

Pumas-AI Products

Pumas-CP

For individuals looking to perform non-compartmental, bioequivalence, and superposition analyses via graphical user interface. Available via desktop or cloud.

Pumas

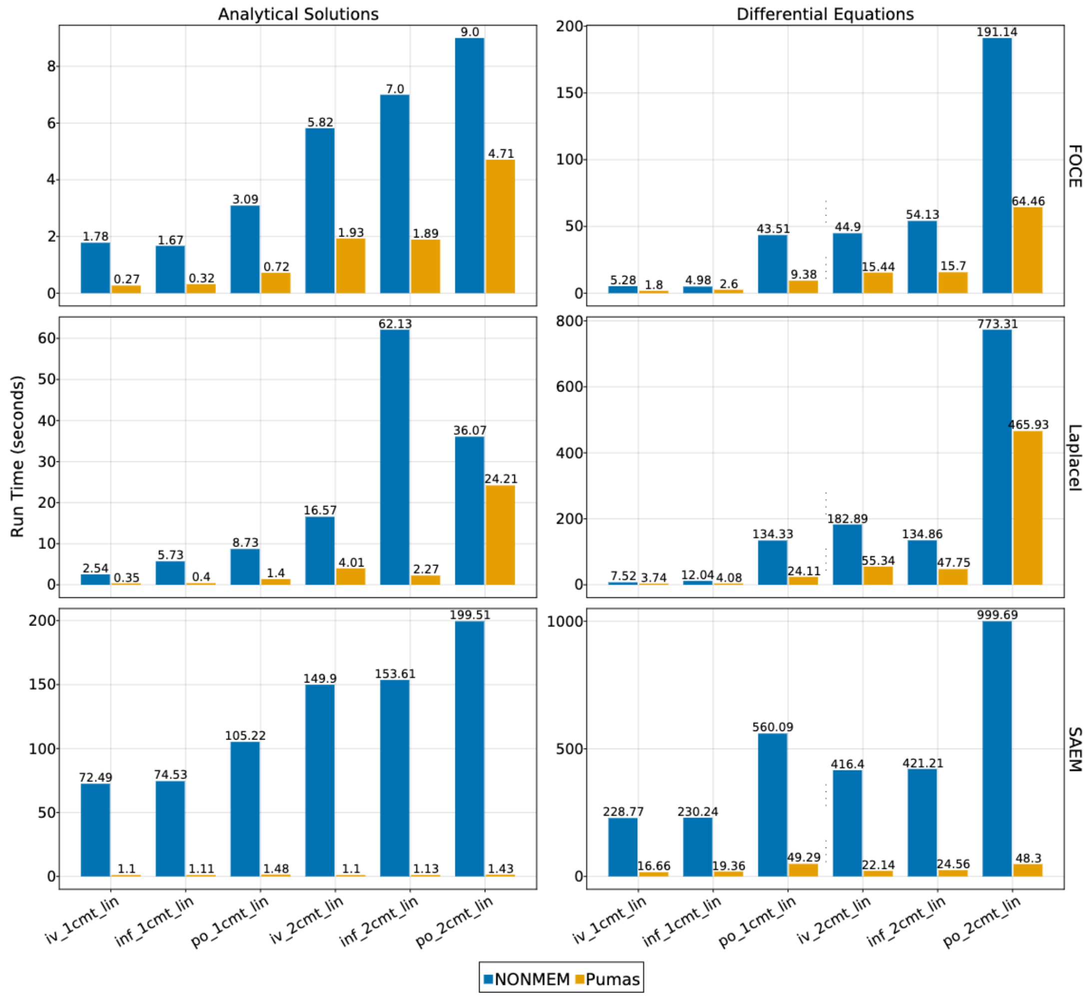
For individuals looking to perform non-compartmental, bioequivalence, nonlinear mixed-effects modeling & simulations, physiologically-based pharmacokinetic and quantitative systems pharmacology modeling. Available via desktop or cloud.

Lyv

For individuals looking for a clinical decision support system leveraging individual patient data for personalized patient care. Available for integration in EHRs such as EPIC and Cerner.

Double Your Think Time

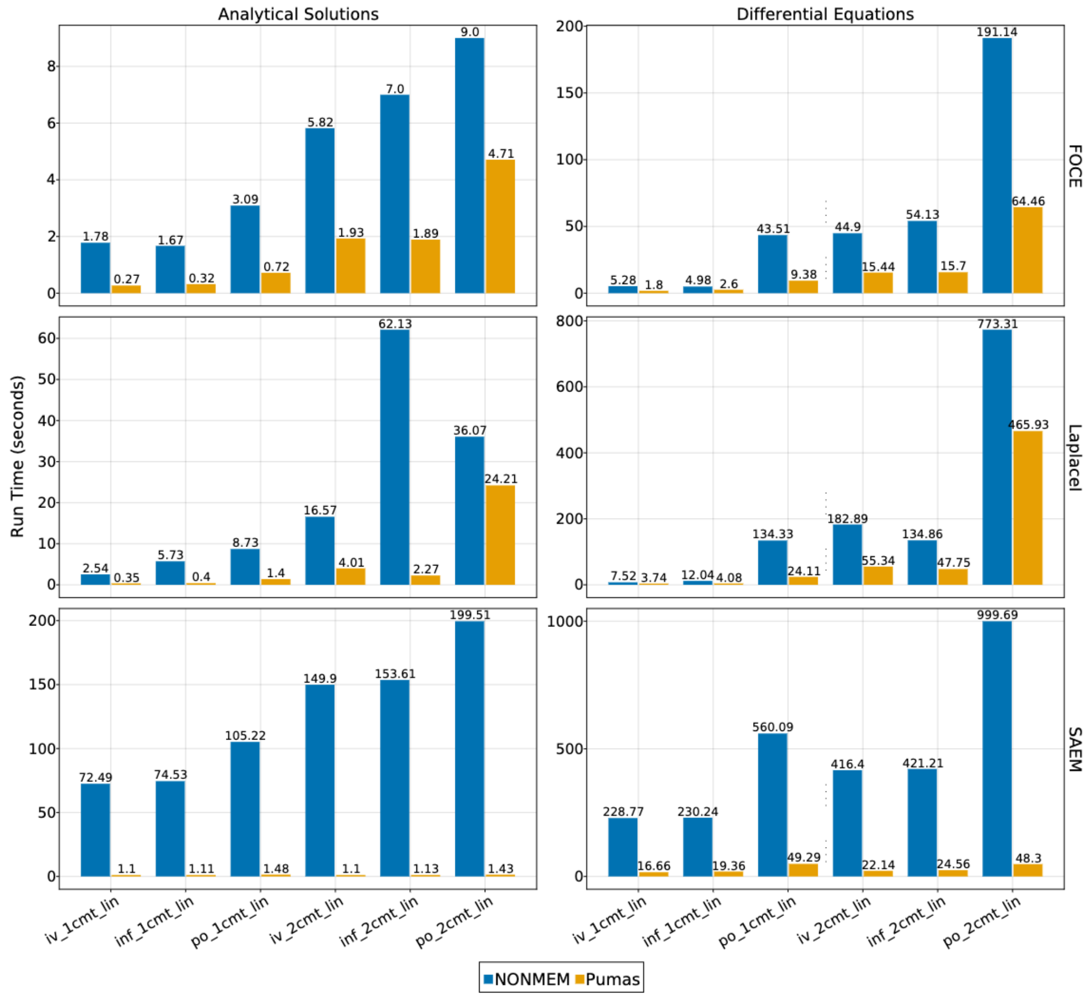
- Used in regulatory submissions in the US, Europe, Asia and Australia, e.g:
  - The Moderna COVID vaccine approvals
- Taught in a number of universities, e.g:
  - University of Maryland
  - Buffalo University
  - University of Florida
- Easier to use and faster than NONMEM
- Has FOCE, Laplace, SAEM and full Bayesian workflows



pumas[AI]

Double Your Think Time

- Free for academics
- Has cloud and desktop versions
- Forum: https://discourse.pumas.ai
- Education: https://pumas.ai/pumas-labs/
- Workshops in PAGE 2023, Spain
  - Bayesian Workflow in Pumas
  - Scientific Modelling Augmented by Machine Learning in DeepPumas



pumas AI

Bayesian in Pumas

pumas AI

Model and Data Definition

- Intuitive and consistent syntax
  - Single subject and hierarchical models
  - Dynamics and covariates
  - Similar in Bayesian and non-Bayesian
- Supports standard data format
- Automation
  - Linearity detection of ODEs
  - Stiffness detection of ODEs
  - Parallelism over subjects
- Error models
  - Continuous and discrete data
  - Censored and time to event models
  - Many distributions
- Wide range of prior distributions
- Integrated NCA parameters

```
poppk2cpt = @model begin
  @param begin
    tvcl ~ LogNormal(log(10), 0.25)
    tvq ~ LogNormal(log(15), 0.5)
    tvvc ~ LogNormal(log(35), 0.25)
    tvvp ~ LogNormal(log(105), 0.5)
    tvka ~ LogNormal(log(2.5), 1)
    σ ~ Constrained(Cauchy(0, 5), lower = 0.0)
    C ~ LKJ(5, 1.0)
    ω ~ Constrained(
      MvNormal(zeros(5), Diagonal(0.4^2 * ones(5))),
      lower=zeros(5),
    )
  end
  @random begin
    ηstd ~ MvNormal(I(5))
  end
  @pre begin
    η = ω .* (getchol(C).L * ηstd)
    CL = tvcl * exp(η[1])
    Q = tvq * exp(η[2])
    Vc = tvvc * exp(η[3])
    Vp = tvvp * exp(η[4])
    Ka = tvka * exp(η[5])
  end
  @dynamics begin
    Depot'      = -Ka*Depot
    Central'    =  Ka*Depot - (CL+Q)/Vc*Central + Q/Vp*Peripheral
    Peripheral' =                Q/Vc*Central - Q/Vp*Peripheral
  end
  @derived begin
    cp := @. Central / Vc
    dv ~ @. LogNormal(log(cp), σ)
  end
end
```

MCMC in Pumas

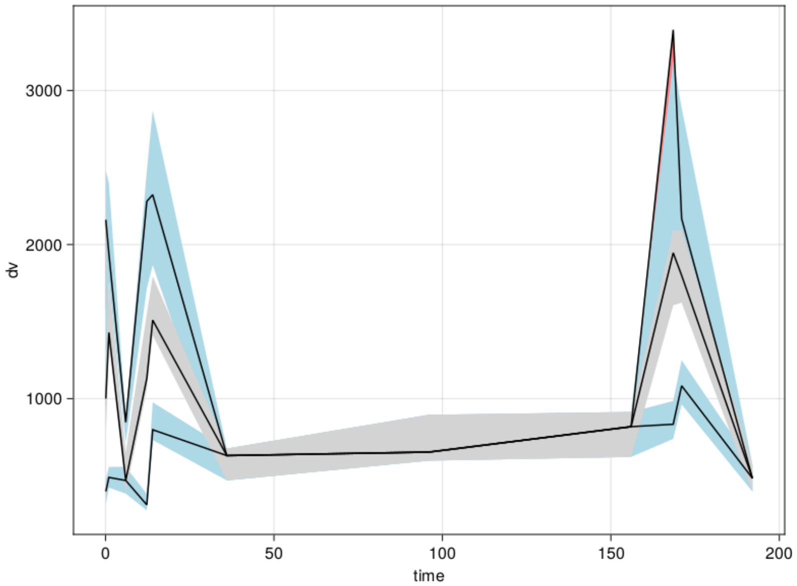
- Accurate and fast
- Joint and marginal posterior sampling
- No-U-Turn Sampler (NUTS) algorithm (used in Stan)
- Easy parallelism over chains and/or over subjects in a population
- Convergence diagnostics and diagnostic plots

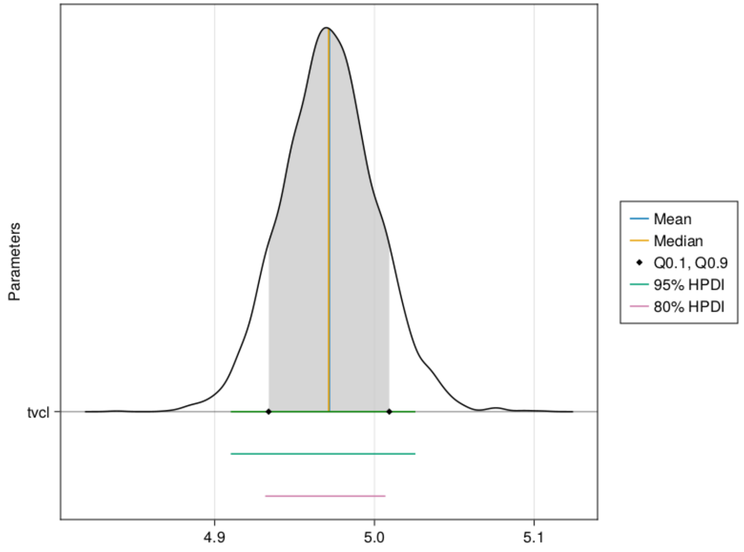
```
# Define algorithm struct
alg = BayesMCMC(
    nsamples = 1_000, # sampling steps
    nadapts = 200, # adaptation steps
    nchains = 4, # number of chains
    ensemblealg = EnsembleThreads(), # multi-threaded
    parallel_chains = true, # parallelism over chains
    parallel_subjects = true, # parallelism over subjects
    time_limit = 600, # in seconds
)
# Call fit
res = fit(model, data, Pumas.init_params(model), alg)

# Or

alg = MarginalMCMC(
    marginal_alg = LaplaceI(),
    nsamples = 1_000,
    nadapts = 200,
    nchains = 4,
    ensemblealg = EnsembleDistributed(), # multi-processing
    parallel_chains = true,
    parallel_subjects = false,
    constantcoef = (σ = 0.1,), # fixing some parameters
)
res = fit(model, data, Pumas.init_params(model), alg)
```

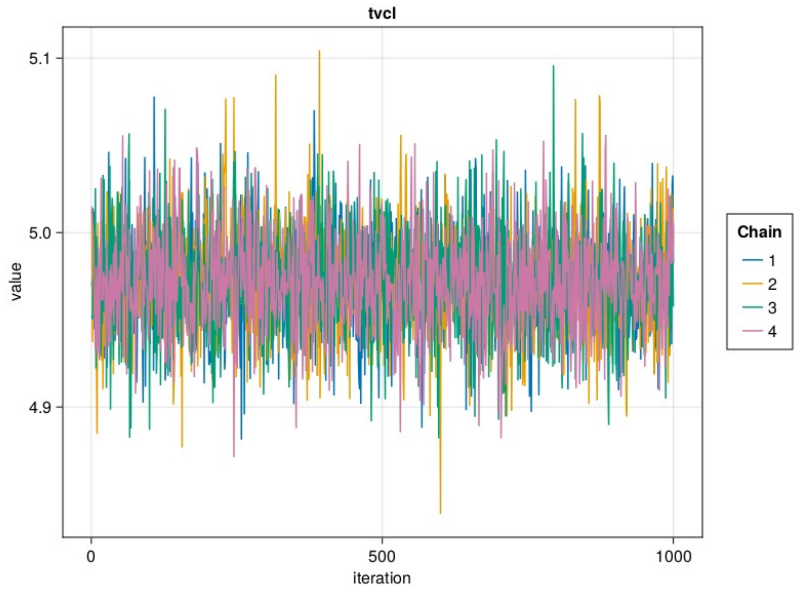
Postprocessing and Simulation

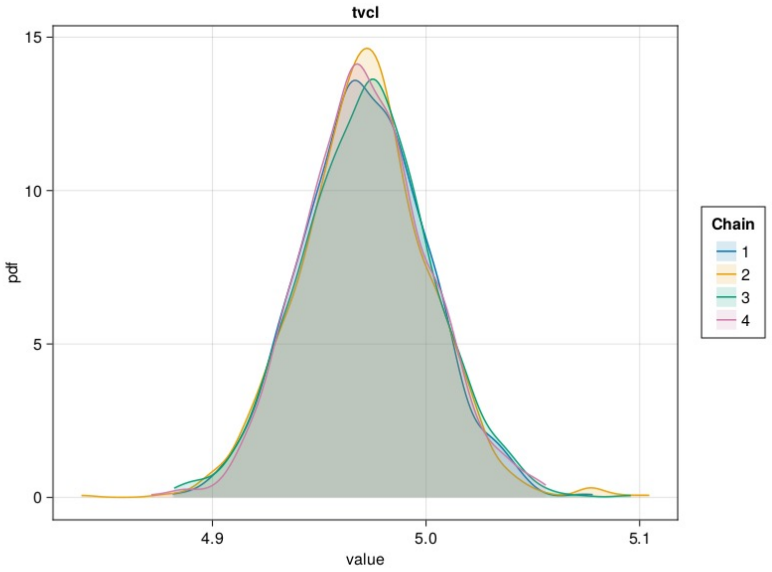
- Simulate/predict using the posterior or prior distributions
- Summary statistics and advanced posterior simulation queries, e.g.
  - Probability that a drug effect is greater than 0
  - Probability that the AUC is lower than a threshold
  - Correlation between individual clearance and volume
  - Credible intervals
- Counter-factual simulations for new drug doses and new subjects
- Model evaluation and cross-validation to detect overfitting

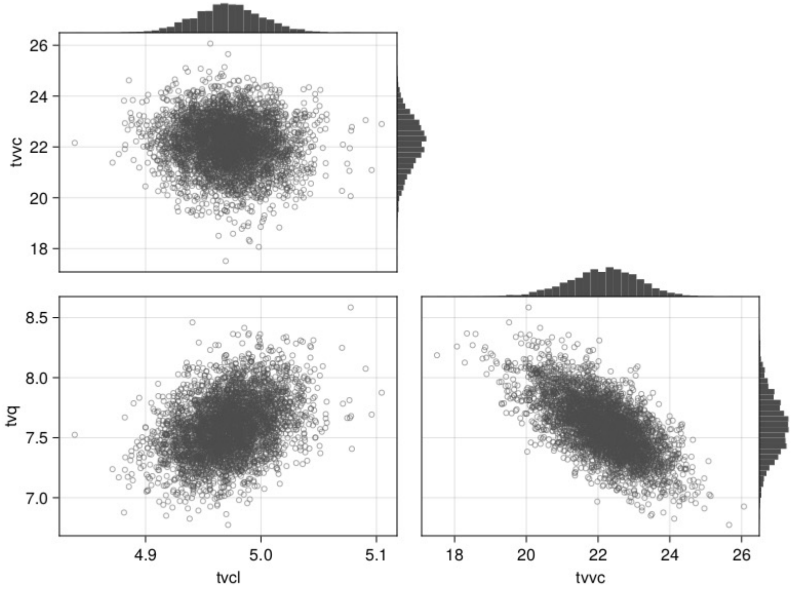
```julia
# Remove warmup and thinning
tres = Pumas.truncate(res; burnin = 1_000, ratio = 0.2)

# Probability of tvcl > 0
mean(tres) do param
  param.tvcl > 0
end

# Probability of prediction > data
mean(sims) do sim, data
  mean(sim.dv .> data.dv)
end

# Diagnostic plots for population or subject-specific parameters
using PumasPlots
trace_plot(tres; parameters = [:tvcl])
trace_plot(tres; subjects = [1, 2])
cummean_plot(tres; parameters = [:tvcl])
density_plot(tres; subjects = [1, 2])
autocor_plot(tres; parameters = [:tvcl])
ridgeline_plot(tres; subject = 1)

# Simulation from the posterior for training subject 1
sims = simobs(tres, subject = 1, samples = 500)
# Simulation from the posterior for a new subject
sims = simobs(tres, new_subject, samples = 500)

# VPC plot
vpc_res = vpc(sims)
vpc_plot(vpc_res)

# PSIS crossvalidation
cv_method = PSISCrossvalidation(LeaveFutureK(K = 1), BySubject())
# or
cv_method = PSISCrossvalidation(LeaveK(K = 1), ByObservation())
cv_res = crossvalidate(tres, cv_method);
# Expected log predictive density
elpd(cv_res)
```

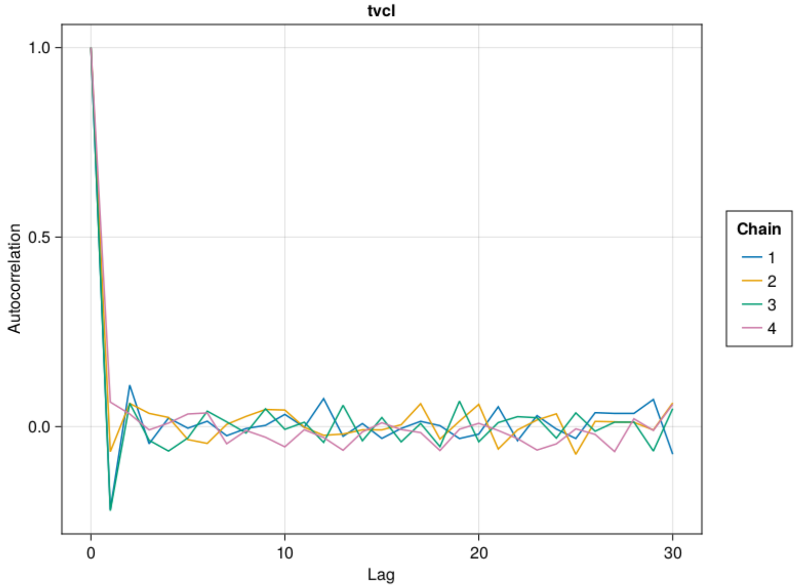

dv
3000
2000
1000
0
50
100
150
200
time


Parameters
tvcl
4.9
5.0
5.1
Mean
Median
Q0.1, Q0.9
95% HPDI
80% HPDI

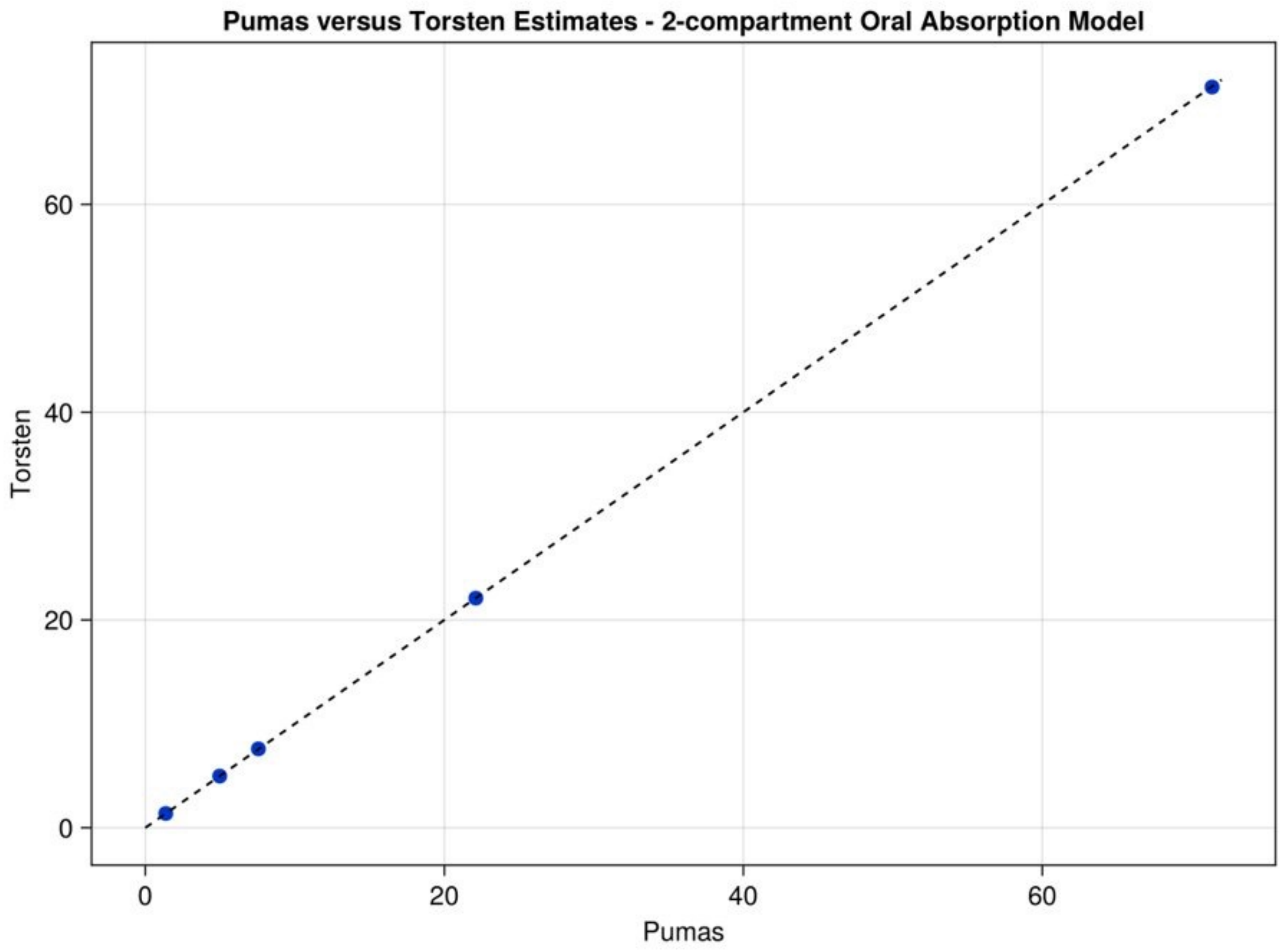

tvcl
value
5.1
5.0
4.9
0
500
1000
iteration
Chain
1
2
3
4


tvcl
pdf
15
10
5
0
4.9
5.0
5.1
value
Chain
1
2
3
4


tvvc
26
24
22
20
18
tvq
8.5
8.0
7.5
7.0
4.9
5.0
5.1
tvcl
18
20
22
24
26
tvvc

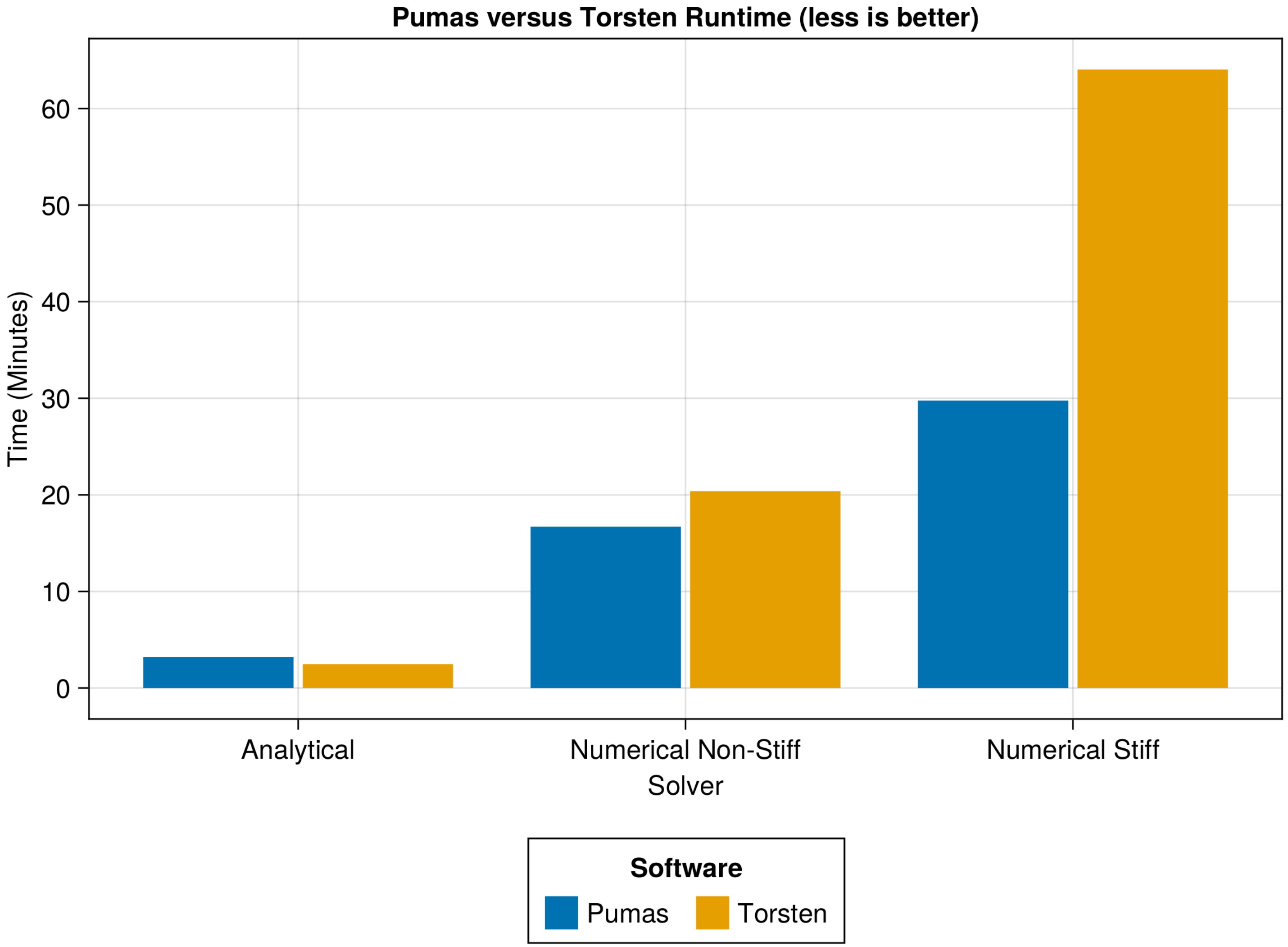

tvcl
Autocorrelation
1.0
0.5
0.0
0
10
20
30
Lag
Chain
1
2
3
4

Pumas vs Torsten (Stan)



Torsten 543 LoC, Pumas 43 LoC

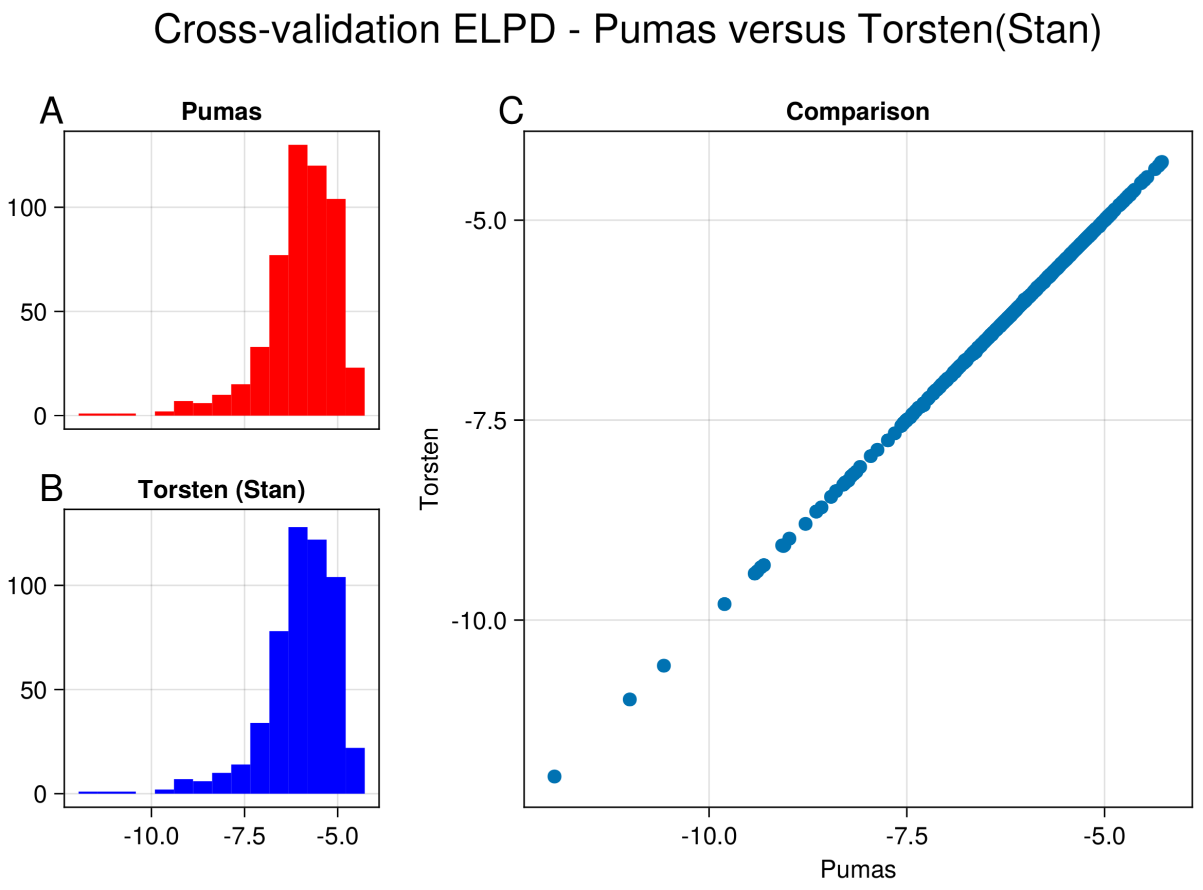
Pumas vs Torsten (Stan)



Torsten 543 LoC, Pumas 43 LoC
Torsten using reduce_sum parallelization

pumas AI

Pumas vs Torsten (Stan)



pumas AI