Lotka-Volterra Parameter Estimation Benchmarks

Vaibhav Dixit, Chris Rackauckas

July 5, 2020

1 Parameter estimation of Lotka Volterra model using optimisation methods

```
using ParameterizedFunctions, OrdinaryDiffEq, DiffEqParamEstim
using BlackBoxOptim, NLopt, Plots, RecursiveArrayTools, QuadDIRECT
```

```
Error: ArgumentError: Package QuadDIRECT not found in current path:
- Run `import Pkg; Pkg.add("QuadDIRECT")` to install the QuadDIRECT package
.
```

```
gr(fmt=:png)
```

```
Plots.GRBackend()
```

```
loc_bounds = Tuple{Float64, Float64}[(0, 5), (0, 5), (0, 5), (0, 5)]
glo_bounds = Tuple{Float64, Float64}[(0, 10), (0, 10), (0, 10), (0, 10)]
loc_init = [1,0.5,3.5,1.5]
glo_init = [5,5,5,5]
```

```
4-element Array{Int64,1}:
 5
 5
 5
 5
```

```
f = @ode_def LotkaVolterraTest begin
    dx = a*x - b*x*y
    dy = -c*y + d*x*y
end a b c d
```

```
(::Main.##WeaveSandBox#317.LotkaVolterraTest{Main.##WeaveSandBox#317.var"##
#ParameterizedDiffEqFunction#337",Main.##WeaveSandBox#317.var"###Parameteri
zedTGradFunction#338",Main.##WeaveSandBox#317.var"###ParameterizedJacobianF
unction#339",Nothing,Nothing,ModelingToolkit.ODESystem}) (generic function
with 1 method)
```

```
u0 = [1.0,1.0]                          #initial values
tspan = (0.0,10.0)
p = [1.5,1.0,3.0,1,0]                   #parameters used, these need to be estimated
from the data
tspan = (0.0, 30.0)                     # sample of 3000 observations over the (0,30)
timespan
prob = ODEProblem(f, u0, tspan,p)
```

```
tspan2 = (0.0, 3.0)                    # sample of 3000 observations over the (0,30)
timespan
prob_short = ODEProblem(f, u0, tspan2,p)
```

```
ODEProblem with uType Array{Float64,1} and tType Float64. In-place: true
timespan: (0.0, 3.0)
u0: [1.0, 1.0]
```

```
dt = 30.0/3000
tf = 30.0
tinterval = 0:dt:tf
t  = collect(tinterval)
```

```
3001-element Array{Float64,1}:
  0.0
  0.01
  0.02
  0.03
  0.04
  0.05
  0.06
  0.07
  0.08
  0.09
  ⋮
 29.92
 29.93
 29.94
 29.95
 29.96
 29.97
 29.98
 29.99
 30.0
```

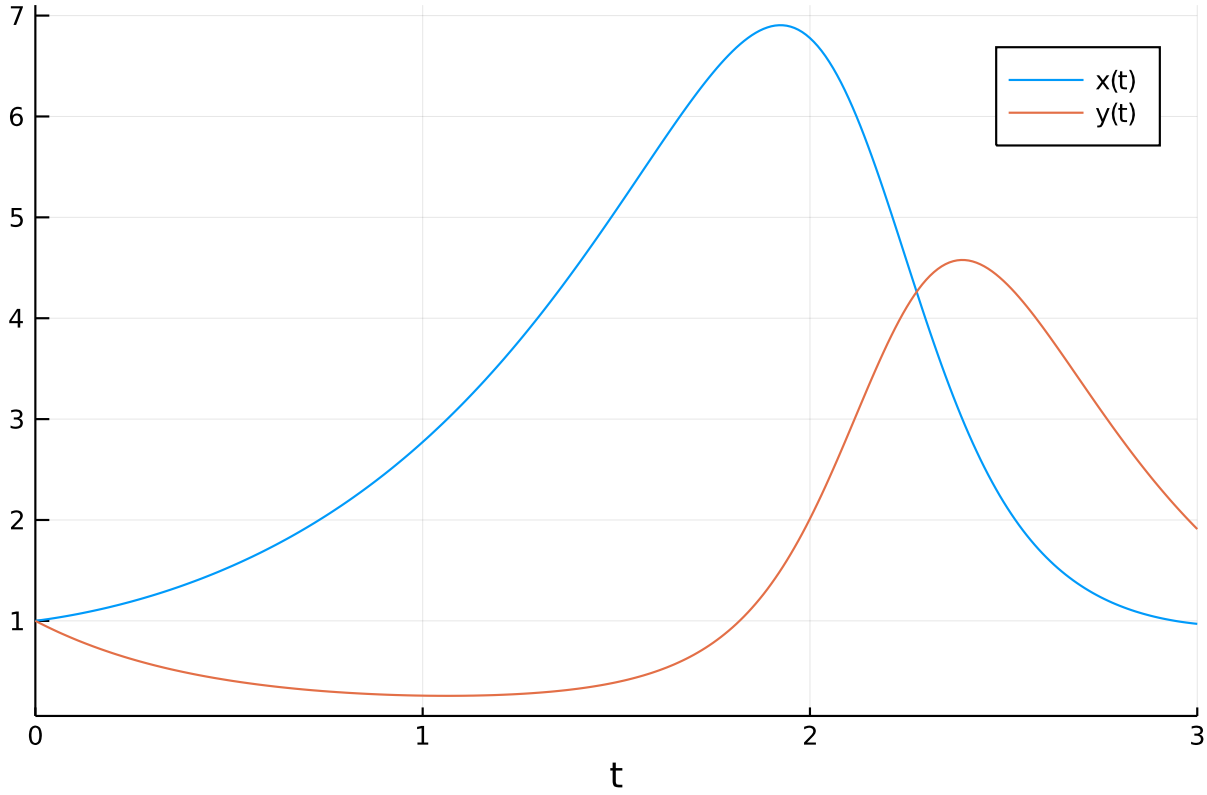
```
h = 0.01
M = 300
tstart = 0.0
tstop = tstart + M * h
tinterval_short = 0:h:tstop
t_short = collect(tinterval_short)
```

```
301-element Array{Float64,1}:
 0.0
 0.01
 0.02
 0.03
 0.04
 0.05
 0.06
 0.07
 0.08
 0.09
 ⋮
 2.92
 2.93
 2.94
 2.95
```

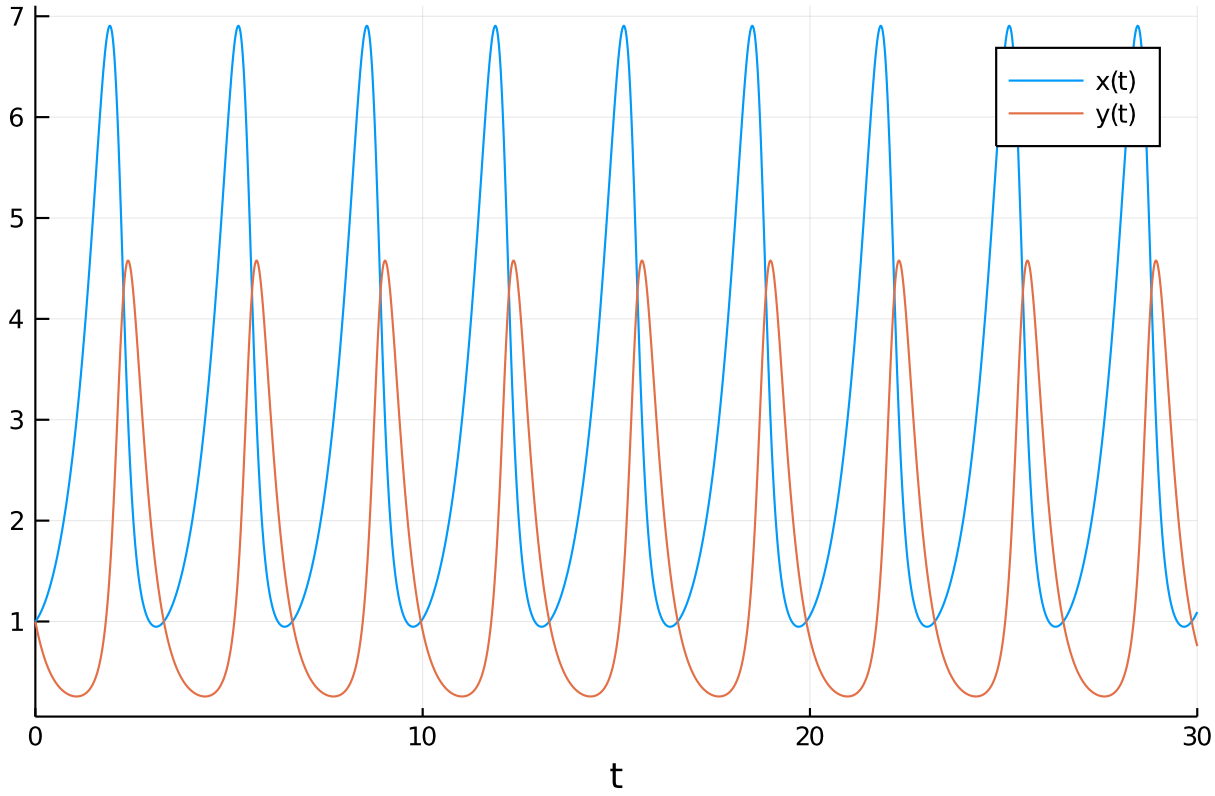
```
 2.96
 2.97
 2.98
 2.99
 3.0

#Generate Data
data_sol_short = solve(prob_short,Tsit5(),saveat=t_short,reltol=1e-9,abstol=1e-9)
data_short = convert(Array, data_sol_short)
data_sol = solve(prob,Tsit5(),saveat=t,reltol=1e-9,abstol=1e-9)
data = convert(Array, data_sol)

2×3001 Array{Float64,2}:
 1.0  1.00511   1.01045   1.01601   1.02179   …   1.07814   1.08595   1.0939
8
 1.0  0.980224  0.960888  0.941986  0.923508      0.785597  0.770673  0.7560
92
```

Plot of the solution

```
p1 = plot(data_sol_short)
```




```
p2 = plot(data_sol)
```



1.0.1 Local Solution from the short data set

```
obj_short =
build_loss_objective(prob_short,Tsit5(),L2Loss(t_short,data_short),tstops=t_short)
res1 = bboptimize(obj_short;SearchRange = glo_bounds, MaxSteps = 7e3)

Starting optimization with optimizer BlackBoxOptim.DiffEvoOpt{BlackBoxOptim
.FitPopulation{Float64},BlackBoxOptim.RadiusLimitedSelector,BlackBoxOptim.A
daptiveDiffEvoRandBin{3},BlackBoxOptim.RandomBound{BlackBoxOptim.Continuous
RectSearchSpace}}
0.00 secs, 0 evals, 0 steps
0.50 secs, 3768 evals, 3656 steps, improv/step: 0.186 (last = 0.1860), fitn
ess=6.319747953

Optimization stopped after 7001 steps and 0.94 seconds
Termination reason: Max number of steps (7000) reached
Steps per second = 7460.37
Function evals per second = 7576.52
Improvements/step = 0.18243
Total function evaluations = 7110


Best candidate found: [1.50218, 1.00146, 2.99401, 0.997438]

Fitness: 0.006008728

# Lower tolerance could lead to smaller fitness (more accuracy)

obj_short =
build_loss_objective(prob_short,Tsit5(),L2Loss(t_short,data_short),tstops=t_short,reltol=1e-9)
```

```
res1 = bboptimize(obj_short;SearchRange = glo_bounds, MaxSteps = 7e3)

Starting optimization with optimizer BlackBoxOptim.DiffEvoOpt{BlackBoxOptim
.FitPopulation{Float64},BlackBoxOptim.RadiusLimitedSelector,BlackBoxOptim.A
daptiveDiffEvoRandBin{3},BlackBoxOptim.RandomBound{BlackBoxOptim.Continuous
RectSearchSpace}}
0.00 secs, 0 evals, 0 steps
0.50 secs, 3739 evals, 3648 steps, improv/step: 0.160 (last = 0.1604), fitn
ess=21.893114770

Optimization stopped after 7001 steps and 0.94 seconds
Termination reason: Max number of steps (7000) reached
Steps per second = 7445.45
Function evals per second = 7541.16
Improvements/step = 0.15829
Total function evaluations = 7091


Best candidate found: [1.50179, 1.00034, 2.99243, 0.996837]

Fitness: 0.006814066

# Change in tolerance makes it worse

obj_short =
build_loss_objective(prob_short,Vern9(),L2Loss(t_short,data_short),tstops=t_short,reltol=1e-9,abstol=1
res1 = bboptimize(obj_short;SearchRange = glo_bounds, MaxSteps = 7e3)

Starting optimization with optimizer BlackBoxOptim.DiffEvoOpt{BlackBoxOptim
.FitPopulation{Float64},BlackBoxOptim.RadiusLimitedSelector,BlackBoxOptim.A
daptiveDiffEvoRandBin{3},BlackBoxOptim.RandomBound{BlackBoxOptim.Continuous
RectSearchSpace}}
0.00 secs, 0 evals, 0 steps
0.50 secs, 2723 evals, 2626 steps, improv/step: 0.178 (last = 0.1778), fitn
ess=364.481754457
1.00 secs, 5709 evals, 5612 steps, improv/step: 0.176 (last = 0.1745), fitn
ess=0.356531822

Optimization stopped after 7001 steps and 1.24 seconds
Termination reason: Max number of steps (7000) reached
Steps per second = 5649.45
Function evals per second = 5726.11
Improvements/step = 0.17314
Total function evaluations = 7096


Best candidate found: [1.50228, 1.00071, 2.9916, 0.996155]

Fitness: 0.013001194

# using the moe accurate Vern9() reduces the fitness marginally and leads to some
increase in time taken
```

2 Using NLopt

```
obj_short =
build_loss_objective(prob_short,Vern9(),L2Loss(t_short,data_short),tstops=t_short,reltol=1e-9,abstol=1

(::DiffEqParamEstim.DiffEqObjective{DiffEqParamEstim.var"#43#48"{Nothing,Bo
ol,Int64,typeof(DiffEqParamEstim.STANDARD_PROB_GENERATOR),Base.Iterators.Pa
irs{Symbol,Any,Tuple{Symbol,Symbol,Symbol},NamedTuple{(:tstops, :reltol, :a
bstol),Tuple{Array{Float64,1},Float64,Float64}}},DiffEqBase.ODEProblem{Arra
y{Float64,1},Tuple{Float64,Float64},true,Array{Float64,1},Main.##WeaveSandB
ox#317.LotkaVolterraTest{Main.##WeaveSandBox#317.var"###ParameterizedDiffEq
Function#337",Main.##WeaveSandBox#317.var"###ParameterizedTGradFunction#338
",Main.##WeaveSandBox#317.var"###ParameterizedJacobianFunction#339",Nothing
,Nothing,ModelingToolkit.ODESystem},Base.Iterators.Pairs{Union{},Union{},Tu
ple{},NamedTuple{(),Tuple{}}},DiffEqBase.StandardODEProblem},OrdinaryDiffEq
.Vern9,DiffEqParamEstim.L2Loss{Array{Float64,1},Array{Float64,2},Nothing,No
thing,Nothing},Nothing},DiffEqParamEstim.var"#47#53"{DiffEqParamEstim.var"#
43#48"{Nothing,Bool,Int64,typeof(DiffEqParamEstim.STANDARD_PROB_GENERATOR),
Base.Iterators.Pairs{Symbol,Any,Tuple{Symbol,Symbol,Symbol},NamedTuple{(:ts
tops, :reltol, :abstol),Tuple{Array{Float64,1},Float64,Float64}}},DiffEqBas
e.ODEProblem{Array{Float64,1},Tuple{Float64,Float64},true,Array{Float64,1},
Main.##WeaveSandBox#317.LotkaVolterraTest{Main.##WeaveSandBox#317.var"###Pa
rameterizedDiffEqFunction#337",Main.##WeaveSandBox#317.var"###Parameterized
TGradFunction#338",Main.##WeaveSandBox#317.var"###ParameterizedJacobianFunc
tion#339",Nothing,Nothing,ModelingToolkit.ODESystem},Base.Iterators.Pairs{U
nion{},Union{},Tuple{},NamedTuple{(),Tuple{}}},DiffEqBase.StandardODEProble
m},OrdinaryDiffEq.Vern9,DiffEqParamEstim.L2Loss{Array{Float64,1},Array{Floa
t64,2},Nothing,Nothing,Nothing},Nothing}}}) (generic function with 2 method
s)

opt = Opt(:GN_ORIG_DIRECT_L, 4)
lower_bounds!(opt,[0.0,0.0,0.0,0.0])
upper_bounds!(opt,[10.0,10.0,10.0,10.0])
min_objective!(opt, obj_short.cost_function2)
xtol_rel!(opt,1e-12)
maxeval!(opt, 10000)
@time (minf,minx,ret) = NLopt.optimize(opt,glo_init)

0.660439 seconds (1.53 M allocations: 277.062 MiB, 3.38% gc time)
(368.38768828453067, [1.7283950617224937, 2.22222222222419, 3.5802469135861
48, 1.1172077427280471], :XTOL_REACHED)

opt = Opt(:GN_CRS2_LM, 4)
lower_bounds!(opt,[0.0,0.0,0.0,0.0])
upper_bounds!(opt,[10.0,10.0,10.0,10.0])
min_objective!(opt, obj_short.cost_function2)
xtol_rel!(opt,1e-12)
maxeval!(opt, 10000)
@time (minf,minx,ret) = NLopt.optimize(opt,glo_init)

1.561918 seconds (2.93 M allocations: 532.285 MiB, 2.75% gc time)
(1.6661075633136318e-16, [1.5000000000712874, 1.0000000000850755, 2.9999999
995024282, 0.9999999999233733], :XTOL_REACHED)

opt = Opt(:GN_ISRES, 4)
lower_bounds!(opt,[0.0,0.0,0.0,0.0])
upper_bounds!(opt,[10.0,10.0,10.0,10.0])
min_objective!(opt, obj_short.cost_function2)
xtol_rel!(opt,1e-12)
maxeval!(opt, 10000)
@time (minf,minx,ret) = NLopt.optimize(opt,glo_init)
```

```
1.693738 seconds (3.90 M allocations: 709.229 MiB, 3.60% gc time)
(0.42244109455194784, [1.4954463690200626, 1.0063347224892594, 3.0231128629
43177, 1.0038816295092088], :MAXEVAL_REACHED)
```

```
opt = Opt(:GN_ESCH, 4)
lower_bounds!(opt,[0.0,0.0,0.0,0.0])
upper_bounds!(opt,[10.0,10.0,10.0,10.0])
min_objective!(opt, obj_short.cost_function2)
xtol_rel!(opt,1e-12)
maxeval!(opt, 10000)
@time (minf,minx,ret) = NLopt.optimize(opt,glo_init)
```

```
1.688214 seconds (3.90 M allocations: 709.229 MiB, 3.61% gc time)
(166.7672573349156, [1.14785532837359, 0.8871602726440688, 5.81716932162591
2, 1.9938534837889086], :MAXEVAL_REACHED)
```

Now local optimization algorithms are used to check the global ones, these use the local constraints, different intial values and time step

```
opt = Opt(:LN_BOBYQA, 4)
lower_bounds!(opt,[0.0,0.0,0.0,0.0])
upper_bounds!(opt,[5.0,5.0,5.0,5.0])
min_objective!(opt, obj_short.cost_function2)
xtol_rel!(opt,1e-12)
maxeval!(opt, 10000)
@time (minf,minx,ret) = NLopt.optimize(opt,loc_init)
```

```
0.057714 seconds (137.29 k allocations: 24.965 MiB)
(1.6660922429170482e-16, [1.5000000000702367, 1.0000000000848144, 2.9999999
9950844, 0.9999999999254128], :XTOL_REACHED)
```

```
opt = Opt(:LN_NELDERMEAD, 4)
lower_bounds!(opt,[0.0,0.0,0.0,0.0])
upper_bounds!(opt,[5.0,5.0,5.0,5.0])
min_objective!(opt, obj_short.cost_function2)
xtol_rel!(opt,1e-12)
maxeval!(opt, 10000)
@time (minf,minx,ret) = NLopt.optimize(opt,loc_init)
```

```
0.080093 seconds (192.28 k allocations: 34.965 MiB)
(1.6660983656762906e-16, [1.5000000000705307, 1.000000000085224, 2.99999999
9507768, 0.9999999999249004], :XTOL_REACHED)
```

```
opt = Opt(:LD_SLSQP, 4)
lower_bounds!(opt,[0.0,0.0,0.0,0.0])
upper_bounds!(opt,[5.0,5.0,5.0,5.0])
min_objective!(opt, obj_short.cost_function2)
xtol_rel!(opt,1e-12)
maxeval!(opt, 10000)
@time (minf,minx,ret) = NLopt.optimize(opt,loc_init)
```

```
0.074028 seconds (204.47 k allocations: 24.405 MiB)
(4.1924234228369356e-16, [1.499999999682718, 1.0000000001713023, 3.00000000
2088545, 1.0000000007284087], :XTOL_REACHED)
```

```
opt = Opt(:LN_COBYLA, 4)
lower_bounds!(opt,[0.0,0.0,0.0,0.0])
upper_bounds!(opt,[5.0,5.0,5.0,5.0])
min_objective!(opt, obj_short.cost_function2)
xtol_rel!(opt,1e-12)
maxeval!(opt, 10000)
@time (minf,minx,ret) = NLopt.optimize(opt,loc_init)
```

```
1.698281 seconds (3.90 M allocations: 709.229 MiB, 3.78% gc time)
(2.095834367163097e-10, [1.4999994892072248, 0.999999852720916, 3.000002650
119225, 1.0000008332818766], :MAXEVAL_REACHED)

opt = Opt(:LN_NEWUOA_BOUND, 4)
lower_bounds!(opt,[0.0,0.0,0.0,0.0])
upper_bounds!(opt,[5.0,5.0,5.0,5.0])
min_objective!(opt, obj_short.cost_function2)
xtol_rel!(opt,1e-12)
maxeval!(opt, 10000)
@time (minf,minx,ret) = NLopt.optimize(opt,loc_init)

0.153277 seconds (88.93 k allocations: 16.171 MiB)
(1.974260823257644e-9, [1.5000002678499593, 1.0000012766604875, 3.000001108
2750206, 1.0000001048380573], :SUCCESS)

opt = Opt(:LN_PRAXIS, 4)
lower_bounds!(opt,[0.0,0.0,0.0,0.0])
upper_bounds!(opt,[5.0,5.0,5.0,5.0])
min_objective!(opt, obj_short.cost_function2)
xtol_rel!(opt,1e-12)
maxeval!(opt, 10000)
@time (minf,minx,ret) = NLopt.optimize(opt,loc_init)

0.034616 seconds (83.86 k allocations: 15.249 MiB)
(1.666235080324952e-16, [1.500000000066557, 1.0000000000838902, 2.999999999
5296456, 0.9999999999316851], :SUCCESS)

opt = Opt(:LN_SBPLX, 4)
lower_bounds!(opt,[0.0,0.0,0.0,0.0])
upper_bounds!(opt,[5.0,5.0,5.0,5.0])
min_objective!(opt, obj_short.cost_function2)
xtol_rel!(opt,1e-12)
maxeval!(opt, 10000)
@time (minf,minx,ret) = NLopt.optimize(opt,loc_init)

1.685028 seconds (3.90 M allocations: 709.229 MiB, 3.03% gc time)
(3.857624483455275e-12, [1.4999999301104459, 0.9999999814386067, 3.00000035
3466034, 1.0000001103210938], :MAXEVAL_REACHED)

opt = Opt(:LD_MMA, 4)
lower_bounds!(opt,[0.0,0.0,0.0,0.0])
upper_bounds!(opt,[5.0,5.0,5.0,5.0])
min_objective!(opt, obj_short.cost_function2)
xtol_rel!(opt,1e-12)
maxeval!(opt, 10000)
@time (minf,minx,ret) = NLopt.optimize(opt,loc_init)

3.340989 seconds (7.76 M allocations: 1.378 GiB, 3.35% gc time)
(6.678300918709951e-15, [1.4999999973062776, 0.9999999994973329, 3.00000001
43108423, 1.0000000046124624], :XTOL_REACHED)

opt = Opt(:LD_TNEWTON_PRECOND_RESTART, 4)
lower_bounds!(opt,[0.0,0.0,0.0,0.0])
upper_bounds!(opt,[5.0,5.0,5.0,5.0])
min_objective!(opt, obj_short.cost_function2)
xtol_rel!(opt,1e-12)
maxeval!(opt, 10000)
@time (minf,minx,ret) = NLopt.optimize(opt,loc_init)
```

```
0.069769 seconds (140.33 k allocations: 25.514 MiB, 14.58% gc time)
(4.192393249865537e-16, [1.4999999996827114, 1.0000000001712928, 3.00000000
2088565, 1.0000000007284133], :SUCCESS)
```

2.1 Now the longer problem is solved for a global solution

Vern9 solver with reltol=1e-9 and abstol=1e-9 is used and the dataset is increased to 3000 observations per variable with the same integration time step of 0.01.

```
obj = build_loss_objective(prob,Vern9(),L2Loss(t,data),tstops=t,reltol=1e-9,abstol=1e-9)
res1 = bboptimize(obj;SearchRange = glo_bounds, MaxSteps = 4e3)
```

```
Starting optimization with optimizer BlackBoxOptim.DiffEvoOpt{BlackBoxOptim
.FitPopulation{Float64},BlackBoxOptim.RadiusLimitedSelector,BlackBoxOptim.A
daptiveDiffEvoRandBin{3},BlackBoxOptim.RandomBound{BlackBoxOptim.Continuous
RectSearchSpace}}
0.00 secs, 0 evals, 0 steps
0.50 secs, 312 evals, 225 steps, improv/step: 0.373 (last = 0.3733), fitnes
s=24605.446311878
1.00 secs, 622 evals, 516 steps, improv/step: 0.318 (last = 0.2749), fitnes
s=21563.127036280
1.50 secs, 927 evals, 821 steps, improv/step: 0.268 (last = 0.1836), fitnes
s=21563.127036280
2.00 secs, 1240 evals, 1134 steps, improv/step: 0.233 (last = 0.1406), fitn
ess=20978.321172655
2.51 secs, 1545 evals, 1439 steps, improv/step: 0.208 (last = 0.1180), fitn
ess=20978.321172655
3.01 secs, 1858 evals, 1752 steps, improv/step: 0.188 (last = 0.0927), fitn
ess=14439.139116309
3.51 secs, 2171 evals, 2065 steps, improv/step: 0.172 (last = 0.0831), fitn
ess=14439.139116309
4.01 secs, 2475 evals, 2369 steps, improv/step: 0.155 (last = 0.0428), fitn
ess=10700.112706369
4.51 secs, 2788 evals, 2682 steps, improv/step: 0.142 (last = 0.0383), fitn
ess=10700.112706369
5.01 secs, 3099 evals, 2994 steps, improv/step: 0.136 (last = 0.0897), fitn
ess=10700.112706369
5.51 secs, 3404 evals, 3299 steps, improv/step: 0.130 (last = 0.0656), fitn
ess=10700.112706369
6.01 secs, 3717 evals, 3612 steps, improv/step: 0.130 (last = 0.1374), fitn
ess=9759.094895437
6.52 secs, 4026 evals, 3921 steps, improv/step: 0.128 (last = 0.1003), fitn
ess=6768.919085269

Optimization stopped after 4001 steps and 6.65 seconds
Termination reason: Max number of steps (4000) reached
Steps per second = 601.81
Function evals per second = 617.61
Improvements/step = 0.12875
Total function evaluations = 4106


Best candidate found: [6.28703, 5.68796, 0.600939, 0.218657]

Fitness: 6768.919085269
```

```
BlackBoxOptim.OptimizationResults("adaptive_de_rand_1_bin_radiuslimited", "
Max number of steps (4000) reached", 4001, 1.5939253942414e9, 6.64825010299
6826, BlackBoxOptim.DictChain{Symbol,Any}[BlackBoxOptim.DictChain{Symbol,An
y}[Dict{Symbol,Any}(:RngSeed => 600199,:SearchRange => [(0.0, 10.0), (0.0,
10.0), (0.0, 10.0), (0.0, 10.0)],:MaxSteps => 4000),Dict{Symbol,Any}()],Dic
t{Symbol,Any}(:FitnessScheme => BlackBoxOptim.ScalarFitnessScheme{true}(),:
NumDimensions => :NotSpecified,:PopulationSize => 50,:MaxTime => 0.0,:Searc
hRange => (-1.0, 1.0),:Method => :adaptive_de_rand_1_bin_radiuslimited,:Max
NumStepsWithoutFuncEvals => 100,:RngSeed => 1234,:MaxFuncEvals => 0,:SaveTr
ace => false...)], 4106, BlackBoxOptim.ScalarFitnessScheme{true}(), BlackBoxO
ptim.TopListArchiveOutput{Float64,Array{Float64,1}}(6768.919085268655, [6.2
87033736324417, 5.687958522970391, 0.6009387848703543, 0.21865705845015337]
), BlackBoxOptim.PopulationOptimizerOutput{BlackBoxOptim.FitPopulation{Floa
t64}}(BlackBoxOptim.FitPopulation{Float64}([8.319988850677296 5.35090530320
7101 ... 9.208284892097966 9.178047255746833; 7.626465587731147 5.24516306016
8909 ... 9.087644138710889 8.767429448468071; 3.8860901376078067 0.8945964403
971276 ... 3.608126030852059 1.7544622889540062; 1.9863350832038786 0.4705016
152384405 ... 1.623391391209879 0.7824472876191755], NaN, [23192.382114460386
, 23096.364434682346, 23316.086675737733, 23214.537721685014, 22629.8103014
52024, 23375.725945446142, 23686.938187034502, 23268.61171436174, 12986.774
272839724, 19672.63950066156  ...  21385.074689286783, 21093.18938760146, 235
20.852481472222, 21319.488739072942, 21385.276956722253, 23471.78195160978,
 21332.105432179364, 6981.017939788122, 22978.41674946245, 21902.6298409607
84], 0, BlackBoxOptim.Candidate{Float64}[BlackBoxOptim.Candidate{Float64}([
9.126472801181421, 9.208546165287059, 3.5102994242888528, 1.773423689372020
6], 46, 23471.78195160978, BlackBoxOptim.AdaptiveDiffEvoRandBin{3}(BlackBox
Optim.AdaptiveDiffEvoParameters(BlackBoxOptim.BimodalCauchy(Distributions.C
auchy{Float64}(μ=0.65, σ=0.1), Distributions.Cauchy{Float64}(μ=1.0, σ=0.1),
 0.5, false, true), BlackBoxOptim.BimodalCauchy(Distributions.Cauchy{Float6
4}(μ=0.1, σ=0.1), Distributions.Cauchy{Float64}(μ=0.95, σ=0.1), 0.5, false,
 true), [0.9894622814738603, 0.6058945678006931, 0.5126179966142019, 0.9949
937294675828, 1.0, 0.5917740625370446, 0.9535588066374913, 0.61973945054633
12, 0.9985560455581568, 0.8624129940572194  ...  0.9531769583123025, 0.927129
592559137, 0.6024303527916313, 0.7902996524197038, 1.0, 0.5572237260712672,
 0.6740576854079945, 0.602080022117456, 0.5285032933479916, 0.6563565935585
343], [0.10960211074417733, 1.0, 0.13295007226400213, 1.0, 1.0, 0.776497400
715527, 0.12092130075173885, 0.8899767385976984, 0.08645400832135874, 0.021
09755030824559  ...  0.13011171065094393, 0.23188670335828002, 0.109933742024
14906, 0.939807261374223, 0.023058468666521267, 0.46519998392952644, 1.0, 1
.0, 1.0, 0.9847594606994448])), 0), BlackBoxOptim.Candidate{Float64}([9.126
472801181421, 9.208546165287059, 0.6644160217493473, 1.7734236893720206], 4
6, 39649.314065459366, BlackBoxOptim.AdaptiveDiffEvoRandBin{3}(BlackBoxOpti
m.AdaptiveDiffEvoParameters(BlackBoxOptim.BimodalCauchy(Distributions.Cauch
y{Float64}(μ=0.65, σ=0.1), Distributions.Cauchy{Float64}(μ=1.0, σ=0.1), 0.5
, false, true), BlackBoxOptim.BimodalCauchy(Distributions.Cauchy{Float64}(μ
=0.1, σ=0.1), Distributions.Cauchy{Float64}(μ=0.95, σ=0.1), 0.5, false, tru
e), [0.9894622814738603, 0.6058945678006931, 0.5126179966142019, 0.99499372
94675828, 1.0, 0.5917740625370446, 0.9535588066374913, 0.6197394505463312,
0.9985560455581568, 0.8624129940572194  ...  0.9531769583123025, 0.9271295925
59137, 0.6024303527916313, 0.7902996524197038, 1.0, 0.5572237260712672, 0.6
740576854079945, 0.602080022117456, 0.5285032933479916, 0.6563565935585343]
, [0.10960211074417733, 1.0, 0.13295007226400213, 1.0, 1.0, 0.7764974007155
27, 0.12092130075173885, 0.8899767385976984, 0.08645400832135874, 0.0210975
5030824559  ...  0.13011171065094393, 0.23188670335828002, 0.1099337420241490
6, 0.939807261374223, 0.023058468666521267, 0.46519998392952644, 1.0, 1.0,
1.0, 0.9847594606994448])), 0)]))

opt = Opt(:GN_ORIG_DIRECT_L, 4)
lower_bounds!(opt,[0.0,0.0,0.0,0.0])
upper_bounds!(opt,[10.0,10.0,10.0,10.0])
```

```
min_objective!(opt, obj.cost_function2)
xtol_rel!(opt,1e-12)
maxeval!(opt, 10000)
@time (minf,minx,ret) = NLopt.optimize(opt,glo_init)
```

```
3.383772 seconds (6.48 M allocations: 1.169 GiB, 2.50% gc time)
(23525.885834891702, [8.271604938277504, 7.421124828514532, 7.4028349336932
31, 3.7037037037056706], :XTOL_REACHED)
```

```
opt = Opt(:GN_CRS2_LM, 4)
lower_bounds!(opt,[0.0,0.0,0.0,0.0])
upper_bounds!(opt,[10.0,10.0,10.0,10.0])
min_objective!(opt, obj.cost_function2)
xtol_rel!(opt,1e-12)
maxeval!(opt, 20000)
@time (minf,minx,ret) = NLopt.optimize(opt,glo_init)
```

```
43.426324 seconds (31.53 M allocations: 5.697 GiB, 0.95% gc time)
(1.270751747281269e-14, [1.4999999994835165, 1.0000000002328804, 3.00000000
17799984, 1.0000000007581764], :XTOL_REACHED)
```

```
opt = Opt(:GN_ISRES, 4)
lower_bounds!(opt,[0.0,0.0,0.0,0.0])
upper_bounds!(opt,[10.0,10.0,10.0,10.0])
min_objective!(opt, obj.cost_function2)
xtol_rel!(opt,1e-12)
maxeval!(opt, 50000)
@time (minf,minx,ret) = NLopt.optimize(opt,glo_init)
```

```
80.942502 seconds (154.80 M allocations: 27.928 GiB, 2.51% gc time)
(12120.895763240513, [1.231761038166349, 2.7070178333317196, 4.505830258039
168, 1.6053798971907174], :MAXEVAL_REACHED)
```

```
opt = Opt(:GN_ESCH, 4)
lower_bounds!(opt,[0.0,0.0,0.0,0.0])
upper_bounds!(opt,[10.0,10.0,10.0,10.0])
min_objective!(opt, obj.cost_function2)
xtol_rel!(opt,1e-12)
maxeval!(opt, 20000)
@time (minf,minx,ret) = NLopt.optimize(opt,glo_init)
```

```
32.391982 seconds (61.92 M allocations: 11.171 GiB, 2.52% gc time)
(7490.912810348923, [0.7798297923061335, 0.663896359286747, 7.1966539295479
33, 3.161994682140639], :MAXEVAL_REACHED)
```

```
opt = Opt(:LN_BOBYQA, 4)
lower_bounds!(opt,[0.0,0.0,0.0,0.0])
upper_bounds!(opt,[5.0,5.0,5.0,5.0])
min_objective!(opt, obj.cost_function2)
xtol_rel!(opt,1e-12)
maxeval!(opt, 10000)
@time (minf,minx,ret) = NLopt.optimize(opt,loc_init)
```

```
0.948662 seconds (1.83 M allocations: 338.029 MiB, 1.95% gc time)
(1.2706863918599434e-14, [1.4999999994887927, 1.0000000000235084, 3.00000000
176495, 1.0000000007536436], :SUCCESS)
```

```
opt = Opt(:LN_NELDERMEAD, 4)
lower_bounds!(opt,[0.0,0.0,0.0,0.0])
upper_bounds!(opt,[5.0,5.0,5.0,5.0])
```

```
min_objective!(opt, obj.cost_function2)
xtol_rel!(opt,1e-9)
maxeval!(opt, 10000)
@time (minf,minx,ret) = NLopt.optimize(opt,loc_init)

0.772695 seconds (1.47 M allocations: 271.110 MiB, 2.40% gc time)
(1.7650193766385424e-14, [1.499999999818285, 1.0000000007450542, 3.00000000
0845967, 1.000000000383562], :XTOL_REACHED)

opt = Opt(:LD_SLSQP, 4)
lower_bounds!(opt,[0.0,0.0,0.0,0.0])
upper_bounds!(opt,[5.0,5.0,5.0,5.0])
min_objective!(opt, obj.cost_function2)
xtol_rel!(opt,1e-12)
maxeval!(opt, 10000)
@time (minf,minx,ret) = NLopt.optimize(opt,loc_init)

4.691296 seconds (2.30 M allocations: 424.057 MiB, 0.79% gc time)
(21569.713608932932, [3.259711714502602, 2.7238409562806836, 0.858170492661
9689, 0.4044968406585042], :XTOL_REACHED)
```

```
obj_short =
build_loss_objective(prob_short,Tsit5(),L2Loss(t_short,data_short),tstops=t_short)
lower = [0.0,0.0,0.0,0.0]
upper = [5.0,5.0,5.0,5.0]
splits = ([0.0,1.0,3.0],[0.0,1.0,3.0],[0.0,1.0,3.0],[0.0,1.0,3.0])
root, x0 = analyze(obj_short,splits,lower,upper)

Error: UndefVarError: analyze not defined

minimum(root)

Error: UndefVarError: root not defined

obj = build_loss_objective(prob,Vern9(),L2Loss(t,data),tstops=t,reltol=1e-9,abstol=1e-9)
lower = [0.0,0.0,0.0,0.0]
upper = [10.0,10.0,10.0,10.0]
splits = ([0.0,3.0,6.0],[0.0,3.0,6.0],[0.0,3.0,6.0],[0.0,3.0,6.0])
root, x0 = analyze(obj,splits,lower,upper)

Error: UndefVarError: analyze not defined

minimum(root)

Error: UndefVarError: root not defined
```

Parameter estimation on the longer sample proves to be extremely challenging for some of the global optimizers. A few give the accurate values, BlacBoxOptim also performs quite well while others seem to struggle with accuracy a lot.

3 Conclusion

In general we observe that lower tolerance lead to higher accuracy but too low tolerance could affect the convergance time drastically. Also fitting a shorter timespan seems to be easier in comparision (quite intutively). NLOpt methods seem to give great accuracy in the shorter problem with a lot of the algorithms giving 0 fitness, BBO performs very well on it with marginal change with `tol` values. In case of global optimization of the longer problem there is some difference in the perfomance amongst the algorithms with `LD_SLSQP GN_ESCH GN_ISRES GN_ORIG_DIRECT_L` performing among the worse, BBO also gives a bit high fitness in comparison. QuadDIRECT gives accurate results in the case of the shorter problem but doesn't perform very well in the longer problem case.

```
using DiffEqBenchmarks
DiffEqBenchmarks.bench_footer(WEAVE_ARGS[:folder],WEAVE_ARGS[:file])
```

3.1 Appendix

These benchmarks are a part of the DiffEqBenchmarks.jl repository, found at: https://github.com/JuliaDi

To locally run this tutorial, do the following commands:

```
using DiffEqBenchmarks
DiffEqBenchmarks.weave_file("ParameterEstimation","LotkaVolterraParameterEstimation.jmd
```

Computer Information:

```
Julia Version 1.4.2
Commit 44fa15b150* (2020-05-23 18:35 UTC)
Platform Info:
  OS: Linux (x86_64-pc-linux-gnu)
  CPU: Intel(R) Core(TM) i7-9700K CPU @ 3.60GHz
  WORD_SIZE: 64
  LIBM: libopenlibm
  LLVM: libLLVM-8.0.1 (ORCJIT, skylake)
Environment:
  JULIA_DEPOT_PATH = /builds/JuliaGPU/DiffEqBenchmarks.jl/.julia
  JULIA_CUDA_MEMORY_LIMIT = 2147483648
  JULIA_PROJECT = @.
  JULIA_NUM_THREADS = 8
```

Package Information:

```
Status: `/builds/JuliaGPU/DiffEqBenchmarks.jl/benchmarks/ParameterEstimation/Project.to
[6e4b80f9-dd63-53aa-95a3-0cdb28fa8baf] BenchmarkTools 0.5.0
[a134a8b2-14d6-55f6-9291-3336d3ab0209] BlackBoxOptim 0.5.0
[593b3428-ca2f-500c-ae53-031589ec8ddd] CmdStan 6.0.6
[ebbdde9d-f333-5424-9be2-dbf1e9acfb5e] DiffEqBayes 2.16.0
```

```
[1130ab10-4a5a-5621-a13d-e4788d82bd4c] DiffEqParamEstim 1.15.0
[ef61062a-5684-51dc-bb67-a0fcdec5c97d] DiffEqUncertainty 1.4.1
[31c24e10-a181-5473-b8eb-7969acd0382f] Distributions 0.23.4
[bbc10e6e-7c05-544b-b16e-64fede858acb] DynamicHMC 2.1.5
[76087f3c-5699-56af-9a33-bf431cd00edd] NLopt 0.6.0
[1dea7af3-3e70-54e6-95c3-0bf5283fa5ed] OrdinaryDiffEq 5.41.0
[65888b18-ceab-5e60-b2b9-181511a3b968] ParameterizedFunctions 5.3.0
[91a5bcdd-55d7-5caf-9e0b-520d859cae80] Plots 1.5.3
[731186ca-8d62-57ce-b412-fbd966d074cd] RecursiveArrayTools 2.5.0
```