Overcoming long run time for Bayesian integrated fish population models

05 Febr 2020

Etienne Rivot

ESE Ecology and Ecosystem Health

Rennes, France

etienne.rivot@agrocampus-ouest.fr

Hierarchical Models

A burgeoning sector of research, driven by advancements in computing and sampling technologies

Why are HM so useful?

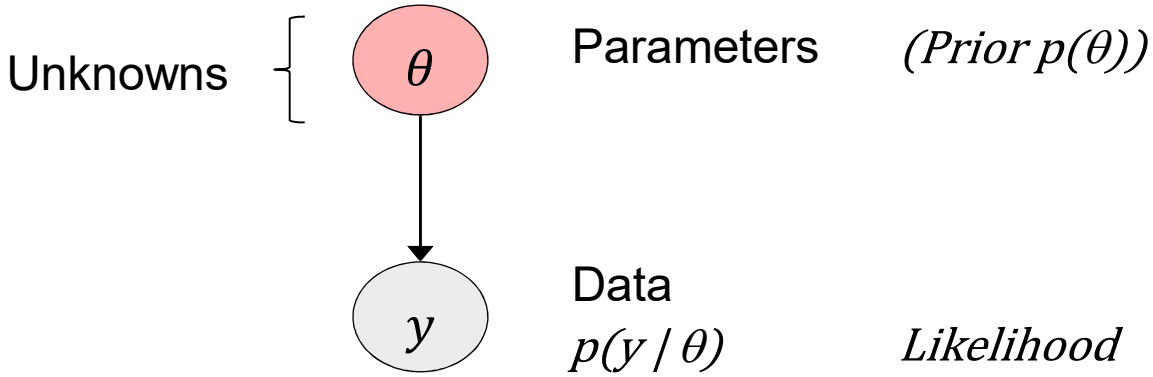
- **HM = "Re-arrange in order"**
- **HM = Models within a model, used to organize the relationships between parameters, variables, and ultimately data**

A wide family of models

Non exhaustive and not exclusive !

- Random effect models
- Observation error models
- State-space models
- HM with Spatial dependencies (*e.g.* CAR …)
- “Integrated models”
- …

Inferences on NON Hierarchical Models



Maximum likelihood

Optimization

$$\hat{\theta} = argmax\ \{p(y|\theta)\}$$

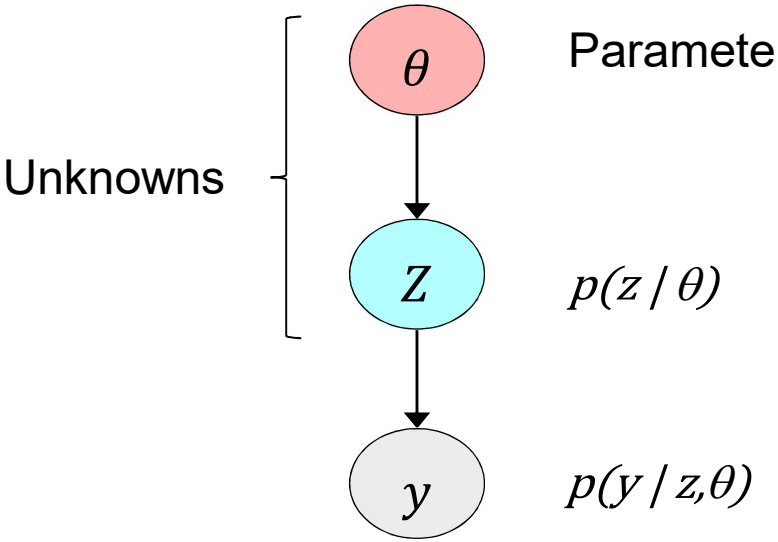
Bayes

Joint posterior distribution

$$p(\theta|y) \propto \underbrace{p(\theta)}_{Param.} \times \underbrace{p(y|\theta)}_{Lik.}$$

Inferences on Hierarchical Models

Clark, 2005 ; Buckland et al., 2007 ; Cressie et al., 2009 ; Parent and Rivot, 2013



All intermediate "process" of interest

z = « latent » or « hidden » variables

Observation process

Likelihood

Maximum likelihood

Latent states must be integrated out

$$\hat{\theta} = argmax\ \{p(y|\theta)\}$$
$$= argmax\left\{\int_{States=z} P(y,z|\theta)dz\right\}$$

Bayes

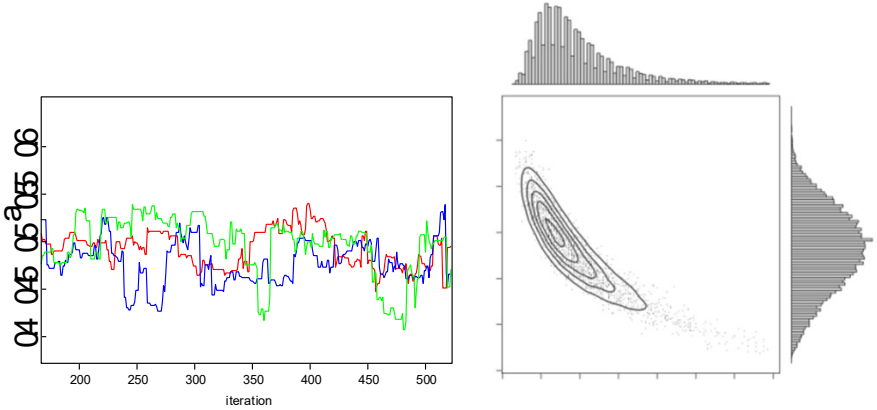
Joint posterior distribution

$$p(\theta,z|y) \propto \underbrace{p(\theta)}_{Param.} \times \underbrace{p(z|\theta)}_{Latent\ var} \times \underbrace{p(y|z,\theta)}_{Lik.}$$

The success story of HBM and BUGS language

The fast and widespread development of H(B)M is intimately linked to the apparition of flexible (and free) softwares

Monte Carlo sampling in posterior distributions



" The genie can not be put back into the bottle. The Bayesian machine, together with MCMC, is arguably the most powerful mechanism never created for processing data and knowledge."

Berger, J. O. (2000). Bayesian analysis: a look at today and thoughts of tomorrow. *Journal of the American Statistical Association*, *95*(452), 1269–1276.

The "*BUGS-boom*" generation

BUGS language:
The most popular engine for Bayesian Hierarchical Modelling

- 

  The original WinBUGS project (first release in 1997)
  Cambridge MRC Biostatistics, Imperial College

  Lunn, D., Spiegelhalter, D., Thomas, A., Best, N. 2009. "The BUGS project: Evolution, critique and future directions". *Statistics in Medicine* **28** (25): 3049–3067.

- **OpenBUGS (2007)**

- **JAGS (2007)**

- 

(2012)

- 

(2014) **…**

Do MCMC freeze the modeler

Malgré la relative facilité de leur mise en oeuvre,

les algorithmes MCMC peuvent parfois être très inefficaces

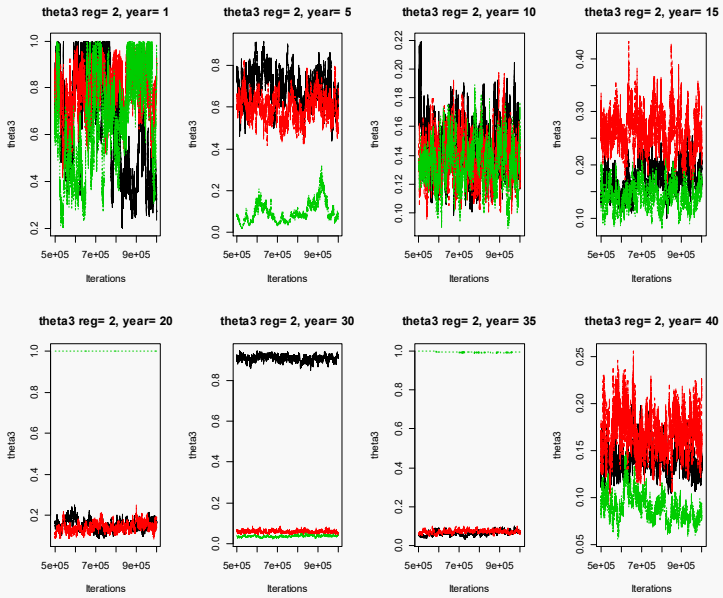
HBM & Fisheries

I love you, neither !

Long run time may be a serious bottleneck

- Bayesian methods are advocated for fisheries stock assessment
- But are still relativey rarely used in practice because of prohibitive run time (~of the order of days to months)

Exploring model sensitivity and evaluating different cases during model development or the review process is difficult



Long run time may be a serious bottleneck

Monnahan, C.C., Branch, T. A., Thorson, J.T., Stewart, I.J., & Szuwalski, C.S. (2019). Overcoming long Bayesian run times in integrated fisheries stock assessments. ICES Journal of Marine Science. https://doi.org/10.1093/icesjms/fsz059

Table 1. Summary of case studies used.

Model name	No. of parameters	Speed (s 1000^{-1} evals)	Brief description	Species and reference	Time needed for ESS = 1000
Hake	217	8.71	MCMC results used for management, empirical weight-at-age, Stock Synthesis	Pacific hake; *Merluccius productus* (Grandin *et al.*, 2016)	18 h
Halibut	195	24.06	Time-varying catchability, empirical weight-at-age, Stock Synthesis	Pacific halibut; *Hippoglossus stenolepis* (Stewart *et al.*, 2016)	12 months
Canary	304	188.10	Time-varying growth, three areas with different exploitation history but no movement, natural mortality varies by age for males, complex selectivity with 31 fleets, Stock Synthesis	Canary rockfish; *Sebastes pinniger* (Thorson and Wetzel, 2015)	187 months
Snow crab	334	18.57	Length-structured, custom built, considerations for sex, maturity state, and shell condition, growth per moult data available	Eastern Bering Sea snow crab; *Chionoecetes opilio* (Szuwalski and Turnock, 2016)	38 months

Speed is how many seconds 1000 model evaluations take and is calculated as warmup and sampling time (but not optimization) divided by the total iterations during a RWM runs in which gradients are not calculated.

Our experience at 

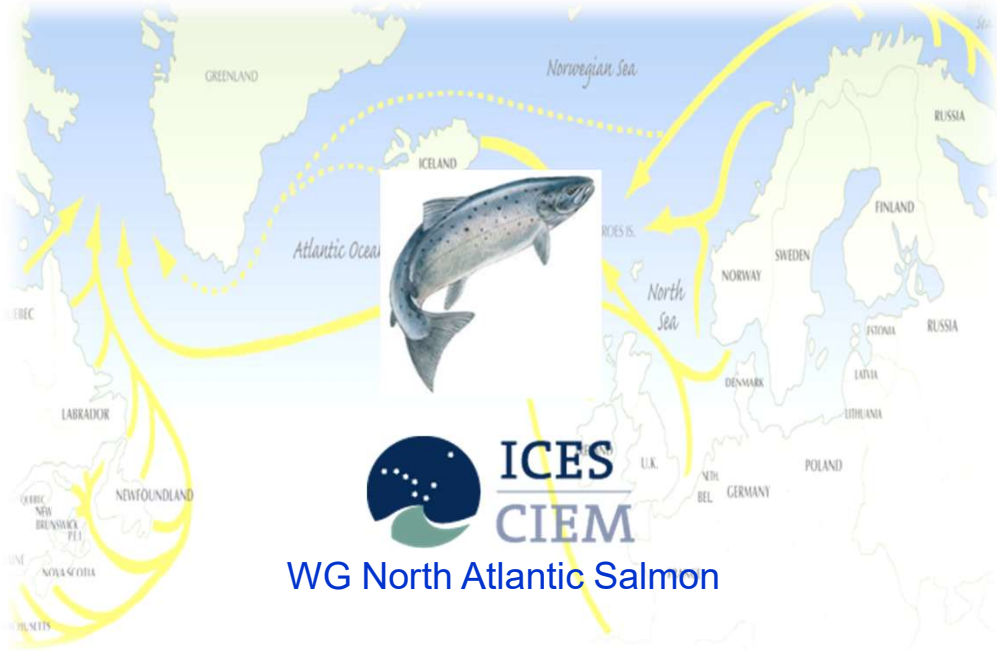
Integrated life cycle model
Common Sole in the Eastern Channel

Rochette, S., Le Pape, O., Vigneau, J., & Rivot, E. (2013). A hierarchical Bayesian model for embedding larval drift and habitat models in integrated life cycles for exploited fish. Ecological Applications, 23(7), 1659–1676.

Archambault, B., Le Pape, O., Baulier, L., Vermard, Y., Véron, M., & Rivot, E. (2016). Adult-mediated connectivity affects inferences on population dynamics and stock assessment of nursery-dependent fish populations. *Fisheries Research*, *181*, 198–213.

Lecomte, J.-B., Le Pape, O., Baillif, H., Nevoux, M., Vermard, Y., Savina-Rolland, M., … Rivot, E. (2019). State-space modeling of multi-decadal mark-recapture data reveals low adult dispersal in a nursery-dependent fish metapopulation. *Canadian Journal of Fisheries and Aquatic Sciences*. https://doi.org/10.1139/cjfas-2019-0037

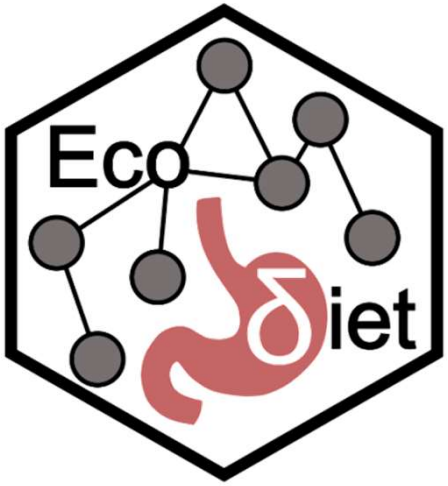
Integrated life cycle model
Atlantic salmon in the North Atlantic Ocean
(basin scale)



Olmos, M., Massiot-Granier, F., Prévost, E., Chaput, G., Bradbury, I. R., Nevoux, M., & Rivot, E. (2019). Evidence for spatial coherence in time trends of marine life history traits of Atlantic salmon in the North Atlantic. *Fish and Fisheries*, *20*(2), 322-342. https://doi.org/10.1111/faf.12345

Olmos, M., Payne, M. R., Nevoux, M., Prévost, E., Chaput, G., Du Pontavice, H., … Rivot, E. (s. d.). Spatial synchrony in the response of a long range migratory species (Salmo salar) to climate change in the North Atlantic Ocean. *Global Change Biology*, *n/a*(n/a). https://doi.org/10.1111/gcb.14913

EcoDiet
A Bayesian integrated approach to build diet matrices integrating information from stomach contents, bio-tracers and the literature



Under review ECOLOGICAL APPLICATIONS ECOLOGICAL SOCIETY OF AMERICA

Integrating information from stomach contents, bio-tracers and the literature to estimate diet matrices

Pierre-Yves Hernvann, Didier Gascuel, Marianne Robert, Dorothée Kopp, Etienne Rivot

Overcoming long Bayesian run time

- Simplify the model by using coarser approximations to the population dynamics
- Forget Bayesian methods and use Optimization approaches (max Likelihood)
- Faster computers
- Parallel MCMC
- Choose inits near the posterior
- Improve priors (regularization, informative priors)
- Parameterization
- Sampling strategy

Overcoming long Bayesian run time

- Différents logiciels (méthodes & algorithmes associés)
  → performances différentes selon la nature du modèle

- Une recherche au cas par cas d'une solution empirique est toujours possible (mais fastidieuse)

- Difficulté

  = identification de la nature du problème (modèle X algorithme)
  & recherche raisonnée d'une solution adaptée

➔ **Groupe de travail Modèle Hiérarchique**

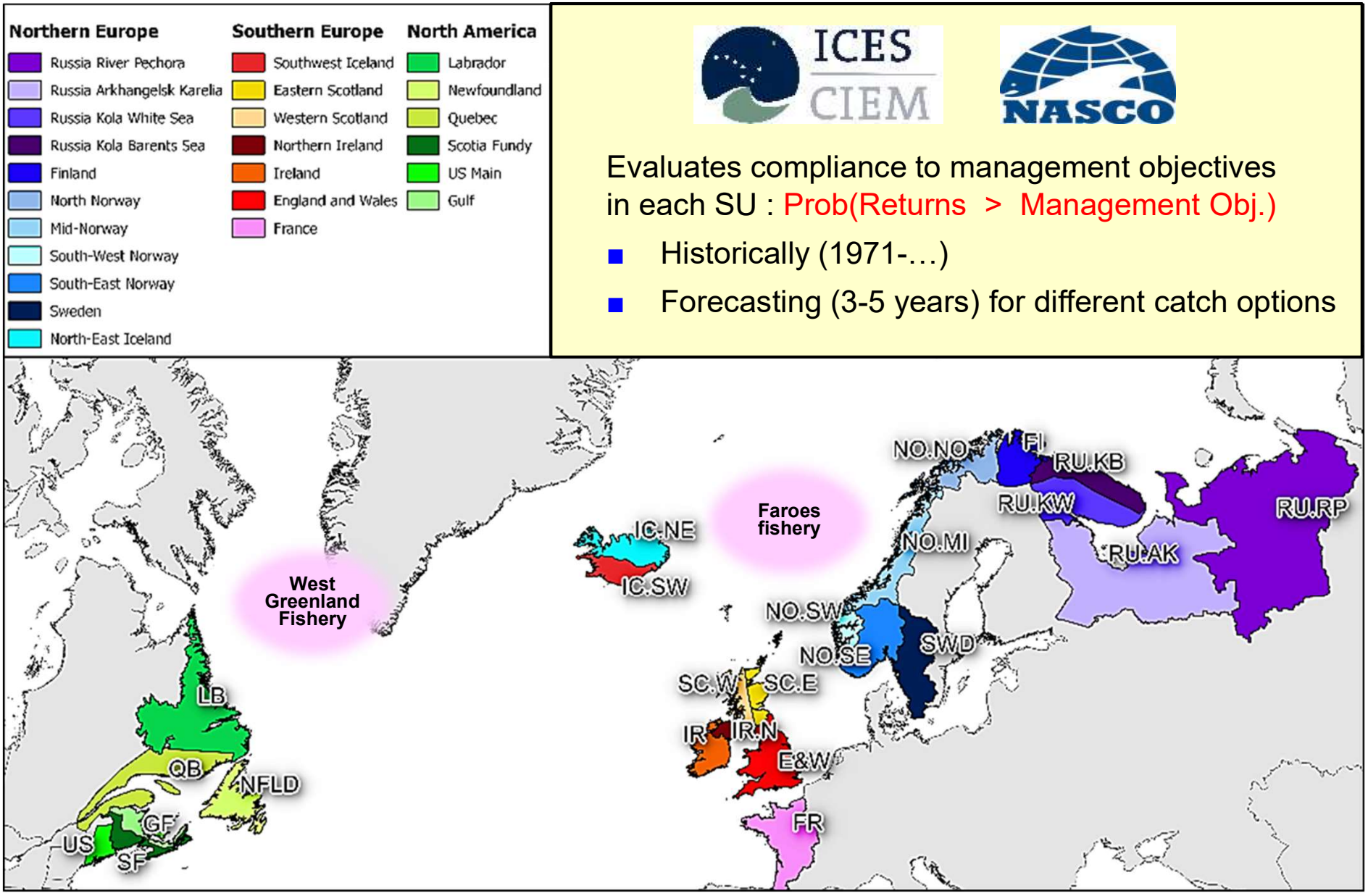
Outlines

- Case study - Atlantic salmon life cycle model

- Explore strategies to overcome long run time

Based on 

Stock assessment

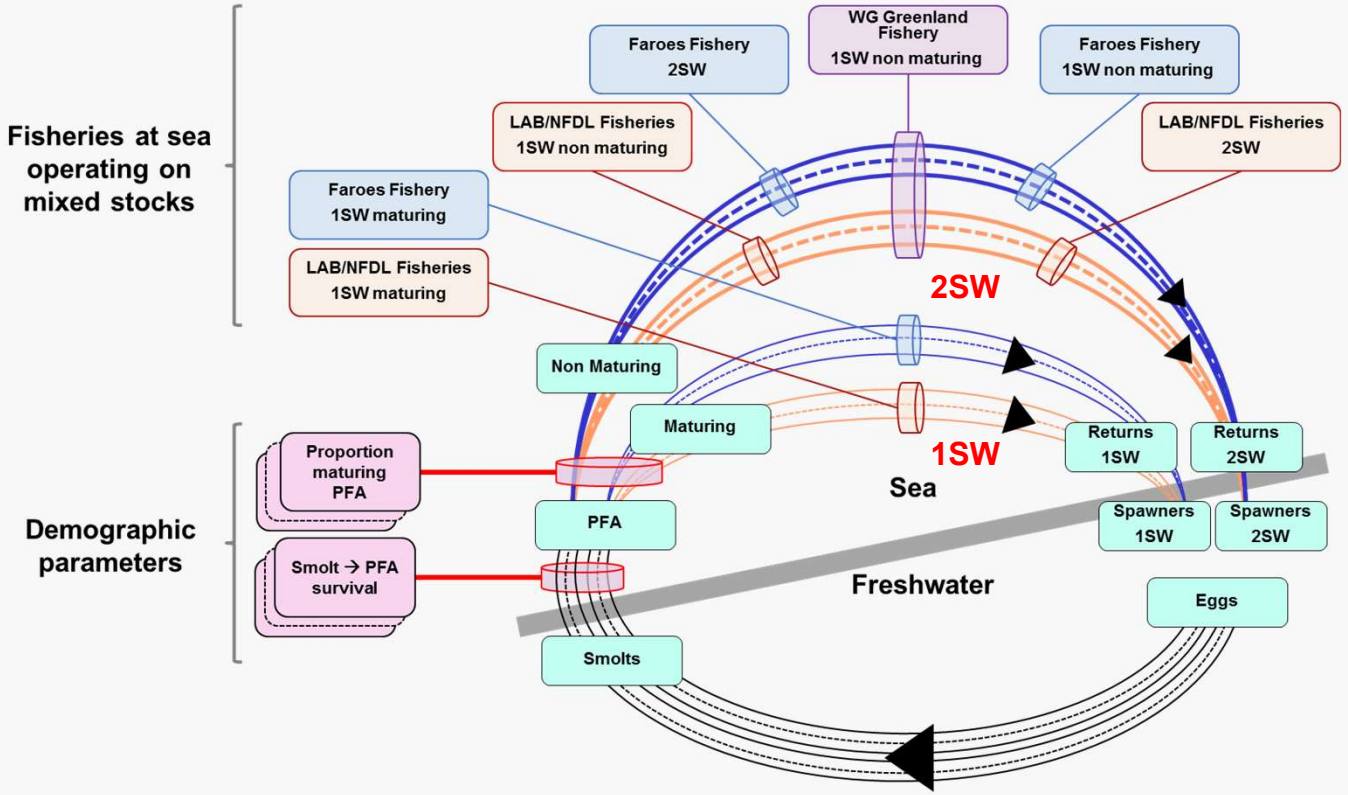


Adapted from Olmos et al., Fish and Fisheries, 2019

One single harmonized (hierarchical) model for the 24 SU

- Assess influences of mixed stock fisheries on returns
- Assess synchronicity in temporal variations of marine survival and prop. maturing

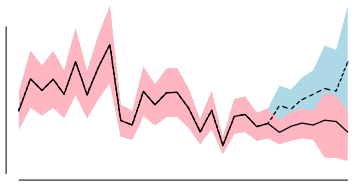
  → Partitioning out synchronous / asynchronous signals



Adapted from Olmos et al., Fish and Fisheries, 2019

JAGS NIMBLE STAN admb

- First written in BUGS, then JAGS, … now in Nimble
- Why Nimble ?
  - Bugs-like code
    - The burden of history of modeler's experience
    - The substantive effort of recoding
    - The inertia of user groups
  - A little bit faster than JAGS
  - Use the same model for estimation / simulation
    - Generate consistent inits
    - The same model object can be used for hindcasting & forecasting

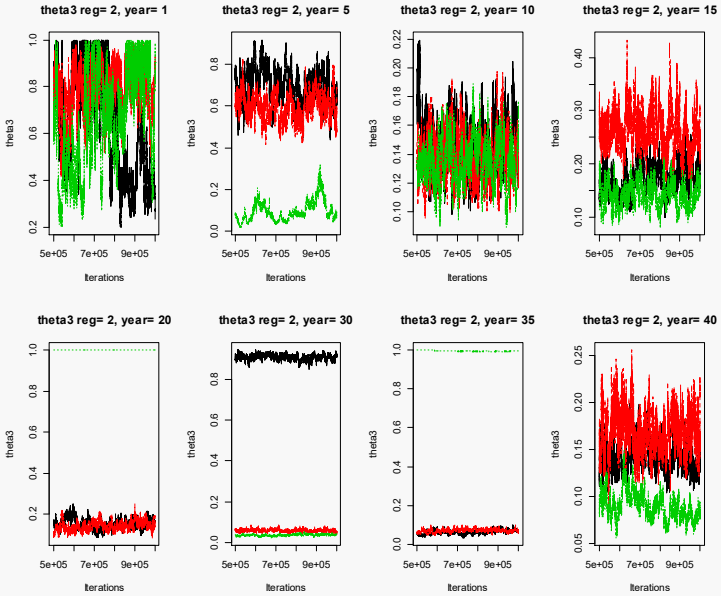


  - Wide possibility to customize sampler (block sampling …)
  - Possibility to use other algorithms on the same model (particles …)

MCMC simulations

- High dimensional space of latent variables (abundances at all life stages x 44 years x 24 stock units)

  Latent variables are all highly correlated



➔ ~ 72 hours (Intel Core i7 – 3.0Ghz)

Comparing different configurations

- Benchmarking
- Baseline version (with good inits)
- Improve parameterization
  - Use deterministic transitions
  - Use customized distributions to integrate out transitions
  - Optimize prior for variance-covariance matrix
- Change default samplers

Comparing different configurations

- Benchmarking
- Baseline version (with good inits)
- Improve parameterization
  - Use deterministic transitions
  - Use customized distributions to integrate out transitions
  - Optimize prior for variance-covariance matrix
- Change default samplers

Comparing different configurations

- **On the same run size**

  - 2 independent chains (parallel cores)
  - n burnin before thin = 50000
  - n thin = 300
  - n posterior samples kept per chain after burnin and after thin = 2000

- **Measuring MCMC efficiency ?**

  Turek et al. 2017
  Monnahan et al. 2017
  Monnahan et al. 2019

  - Algorithmic efficiency
  - Computational efficiency
  - Target nodes (bottlenecks)

Algorithmic efficiency

Turek et al. 2017
Monnahan et al. 2017
Monnahan et al. 2019

- Convergence - Scale Reduction factor (Gelman Rubin) R
- Efficient Sample Size - ESS

ESS = Number of “independent” draws in the posterior

$$ESS = \frac{N_{MCMC}}{\tau}$$

$$\tau = Integrated\ autocorrelation\ time$$

$$\tau = 1 + 2\sum_{k=1}^{\infty} \rho(x_t, x_{t+k})$$

= Number of MCMC sample required for an independent sample to be drawn

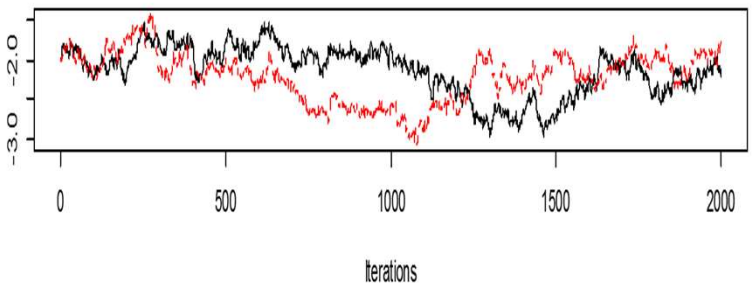
= library(coda), effective.size(), applied to post-burnin and post-thining sample

Efficient sample size

Poor mixing

High autocorrelation

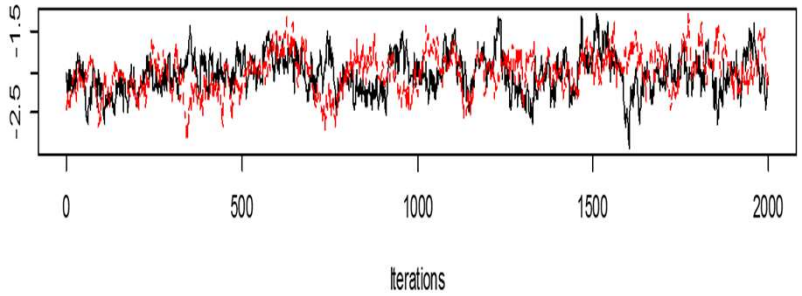
Low $\frac{ESS}{N_{MCMC}}$



Good mixing

Low autocorrelation

High $\frac{ESS}{N_{MCMC}}$



Computational efficiency

Turek et al. 2017
Monnahan et al. 2017
Monnahan et al. 2019

- $= \frac{Alg.efficiency}{run\ time} = \frac{ESS}{run\ time}$

Exclude compilation time but include burnin

- $time_{ESS=1000}$ = Time required to obtain *ESS* = 1000

Identifying bottlenecks

- MCMC often produces good mixing of many nodes, but poor mixing of just a few nodes
- Poorly mixing nodes = bottleneck

➔ Compare models based on performance measured on bottleneck nodes

➔ For each variable, mean of efficiency measure on the 10% “worst” nodes for each variable [24 x 44 years]

Comparing different configurations

- Benchmarking
- Baseline version (with good inits)
- Improve parameterization
  - Use deterministic transitions
  - Use customized distributions to integrate out transitions
  - Optimize prior for variance-covariance matrix
- Change default samplers

Choose initial values near the posterior

- Simulate Nimble model with “good” parameters (from a previous fit) to produce appropriate inits near the posterior

NIMBLE

```
mymod$theta_to_fix  <- value
mymod$simulate(nodes = Nodes_to_simulate)
```

➔ Drastically reduces the number of MCMC draws to be discarded before starting to store them

Bottlenecks

Marine survival and proportion of fish maturing as 1SW

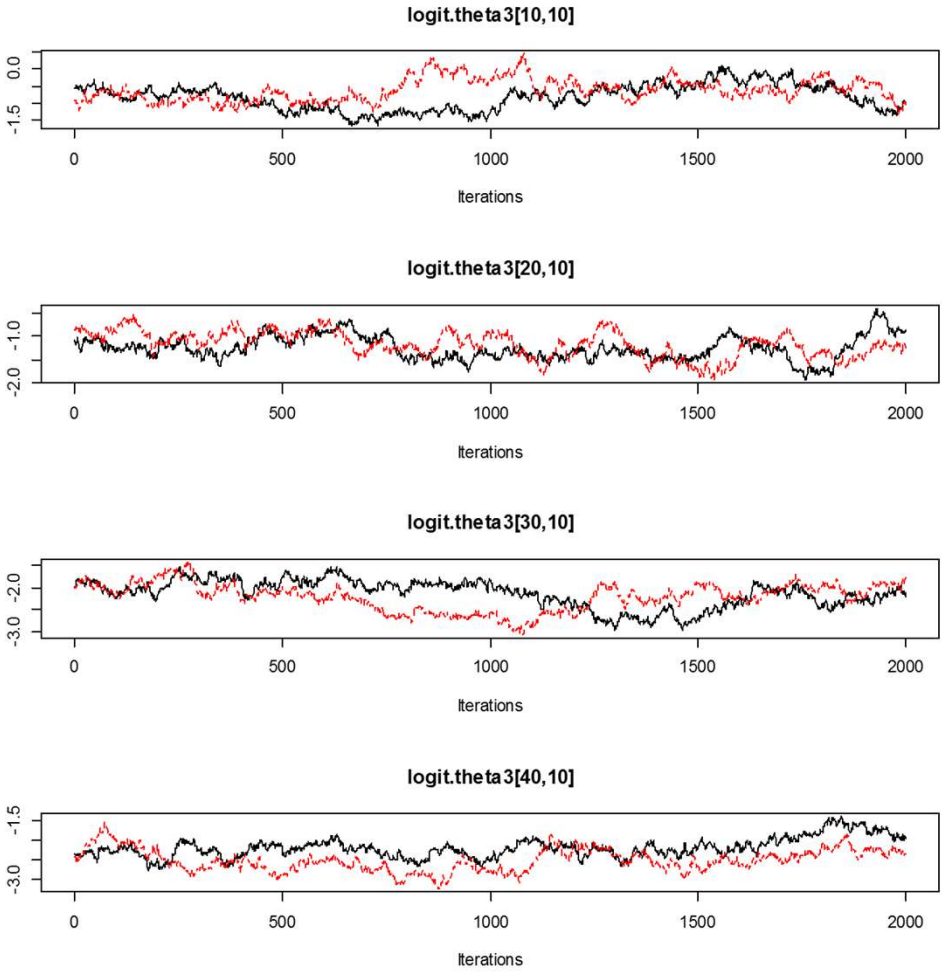
$$logit(\theta_{t+1,1:S}) \sim N^{S}\left(logit(\theta_{t,1:S}), \Sigma\right)$$ **Multi variate random walk**

$$\log(N_{i+1,t+1,s}) \sim N\left(\log(\theta_{i,t,s} \cdot N_{i,t,s}), \sigma\right)$$ **σ fixed to very low value**

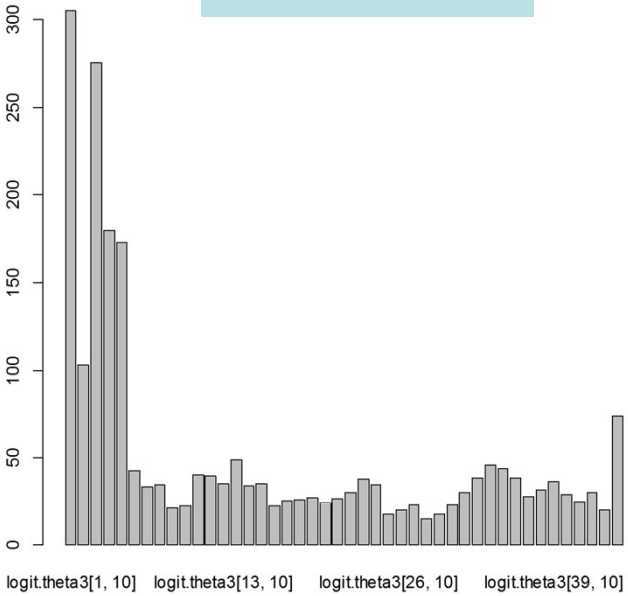
Sequential fisheries (harvest rate *h*)

$$\log(N_{i+1,t,s}) = \log((1 - h_{i,t,s}) \cdot N_{i,t,s})$$

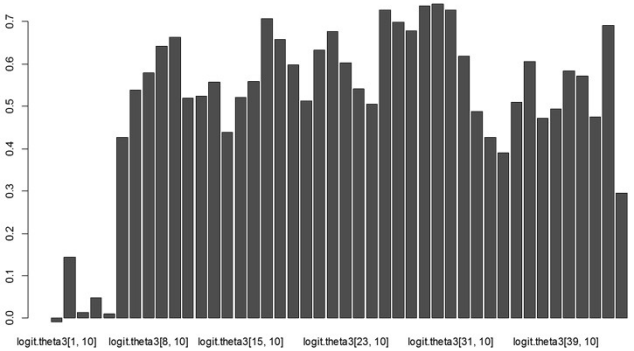
Baseline *V_simple*



ESS

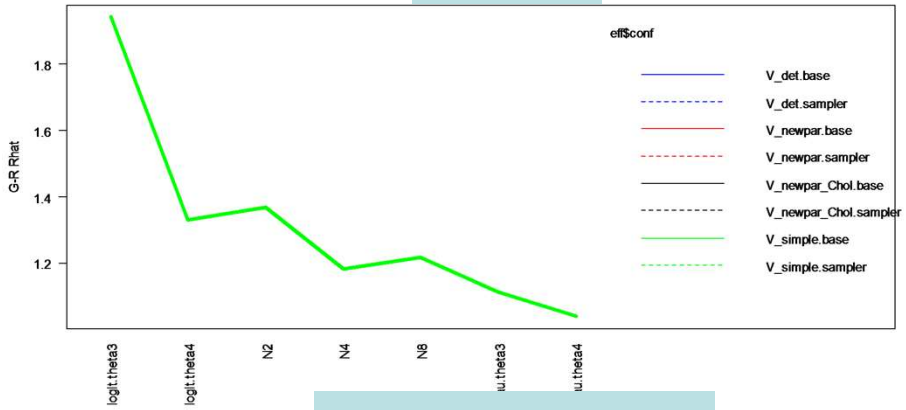


Autocor lag 1

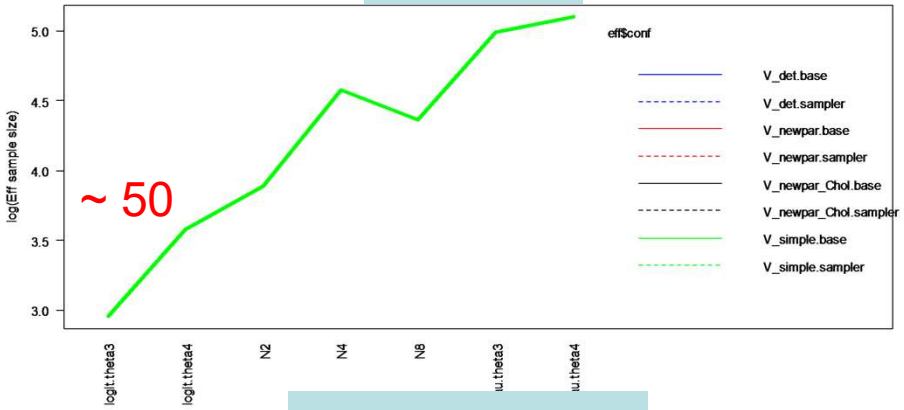


Baseline *V_simple*

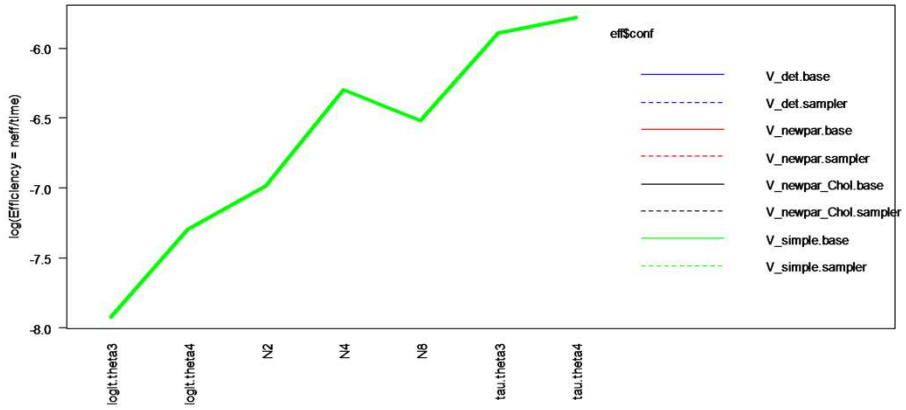
GR R_{hat}



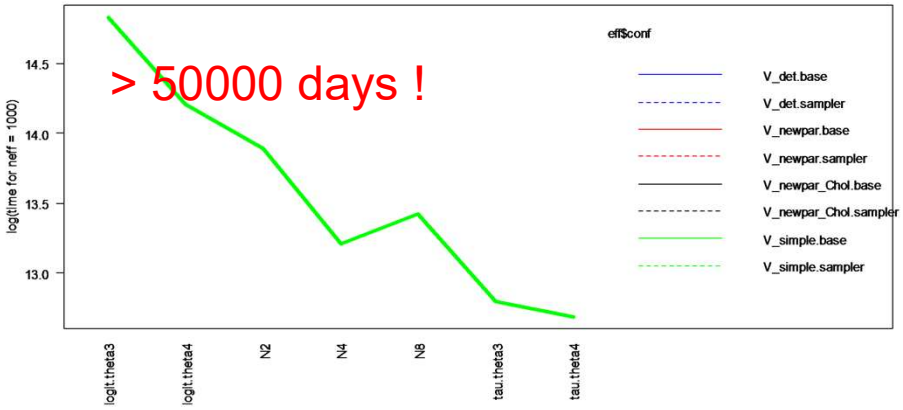
log(ESS)



log(ESS/run time)



log($time_{ESS=1000}$)

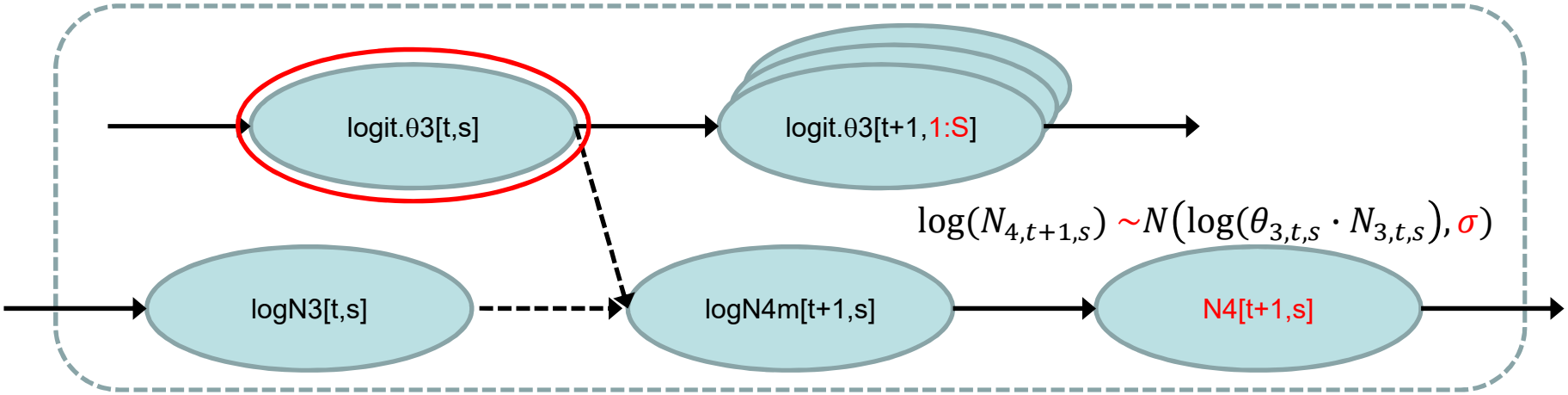


Comparing different configurations

- Benchmarking
- Baseline version (with good inits)
- Improve parameterization
  - Use deterministic transitions
  - Use customized distributions to integrate out transitions
  - Optimize prior for variance-covariance matrix
- Change default samplers

Baseline *V_simple*

➔ Having a lognormal noise with σ fixed to a very low value decreases local dependencies (faster run), but impedes MCMC efficiency



NIMBLE

```
> my.compileNimble$getDependencies(c("logit.theta3[10,1]"))
 "theta3[10, 1]"        "logit.theta3[11, 1:24]"        "log.N4.m[11, 1]"        "N4[11, 1]"
```

Hints and tips

When an MCMC proposal is made for $\theta 3_{t,z}$, say ${\theta 3^*}_{t,z}$, its acceptance as a new MCMC sample depends on the ratio of the conditional probability $[log(N4_{t+1,z})|N3_{t,z}, {\theta 3^*}_{t,z}, \sigma\ ]$ and $[log(N4_{t+1,z})|N3_{t,z}, \theta 3_{t,z}, \sigma\ ]$.

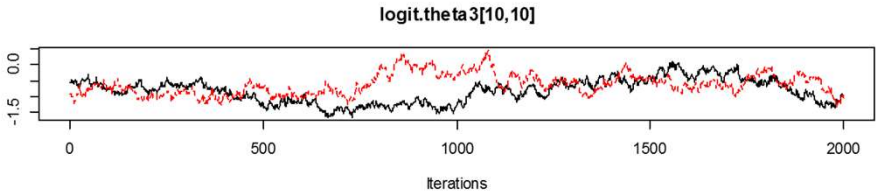
But because σ is very low, only little change is authorized between $\theta 3_{t,z}$ and ${\theta 3^*}_{t,z}$, hence very low mixing.

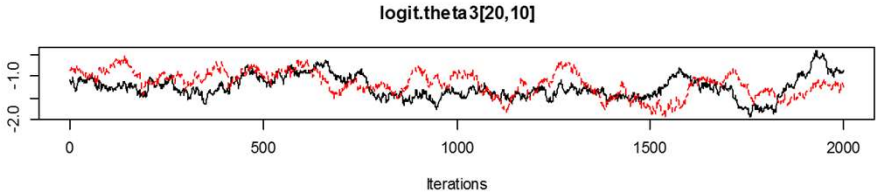
Deterministic *V_det*

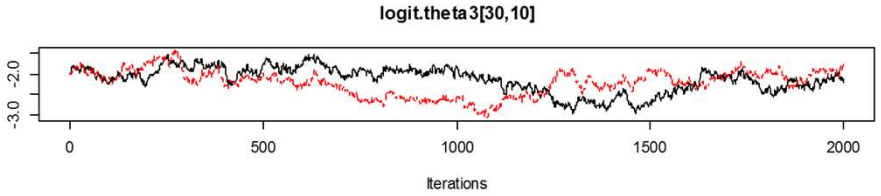
V_simple

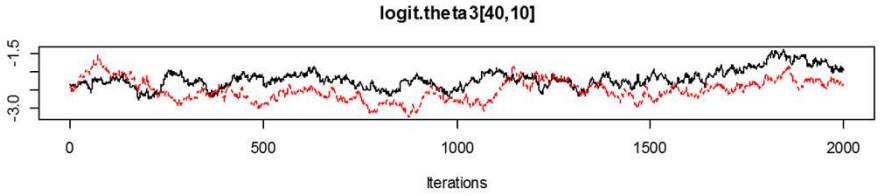
V_det

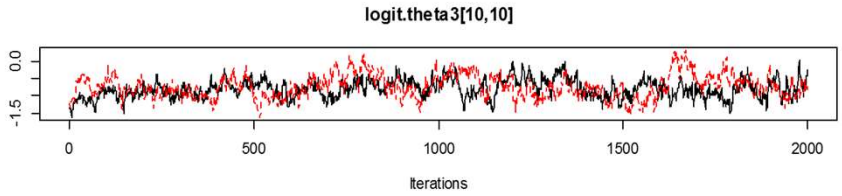
➔ Improves mixing (algorithmic efficiency)

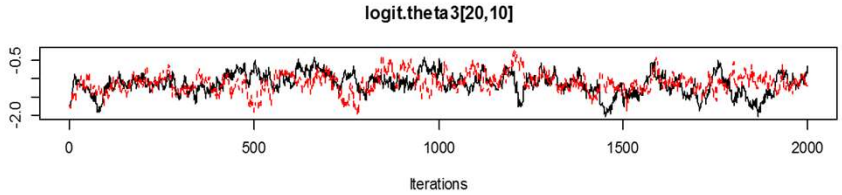


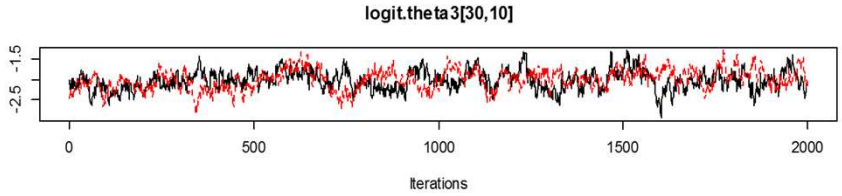


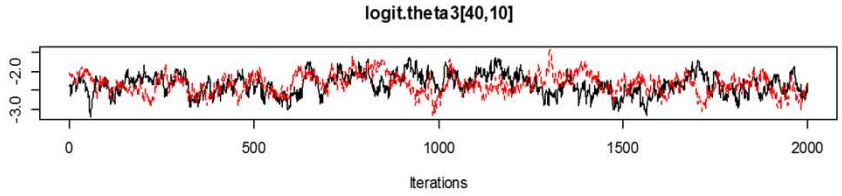












Deterministic *V_det*

➔ But seriously improves computational requirements

NIMBLE

```
> my.compileNimble$getDependencies(c("logit.theta3[10,1]"))
```

[1] "theta3[10, 1]"
[2] "logit.theta3[11, 1:24]"
[3] "N4[11, 1]"
[4] "N5[11, 1]"
[5] "N8[11, 1]"
[6] "C5.NAC.1[11, 1]"
[7] "C5.NAC.2[11, 1]"
[8] "C5.NAC.3[11, 1]"
[9] "N6[11, 1]"
[10] "C8.NAC.1[11, 1]"
[11] "N8.1[11, 1]"
[12] "C5.NAC.2.lab[11]"
[13] "N7[11, 1]"
[14] "Chw.1SW[11, 1]"
[15] "Chw.1SW.delSp[12, 1]"
[16] "lifted_log_oPN6_oBt_comma_r_cB_cP_L223[11, 1]"
[17] "C8.2[11, 1]"
[18] "N8.2[11, 1]"
[19] "C5.NAC.1.tot[11]"
[20] "C5.NAC.3.tot[11]"
[21] "C8.NAC.1.tot[11]"
[22] "lifted_log_oPC5_dot_NAC_dot_2_dot_lab_oBt_cB_cP_L247[11]"
[23] "N1[11, 1]"
[24] "lifted_log_oPChw_dot_1SW_oBt_comma_r_cB_cP_L229[11, 1]"
[25] "log.R1SW.m[11, 1]"
[26] "C8.NAC.3[12, 1]"
[27] "C8.NAC.4[12, 1]"
[28] "C8.NAC.5[12, 1]"
[29] "N9[12, 1]"
[30] "lifted_log_oPC5_dot_NAC_dot_1_dot_tot_oBt_cB_cP_L245[11]"
[31] "lifted_log_oPC5_dot_NAC_dot_3_dot_tot_oBt_cB_cP_L248[11]"
[32] "lifted_log_oPC8_dot_NAC_dot_1_dot_tot_oBt_cB_cP_L250[11]"
[33] "C8.2.tot[11]"
[34] "log.C1.tot.Lb.m[11]"
[35] "log.N2.m[11, 1]"
[36] "log.hwC1SW.m[11, 1]"
[37] "C8.NAC.4.lab[12]"
[38] "N10[12, 1]"
[39] "Chw.2SW[12, 1]"
[40] "Chw.2SW.delSp[13, 1]"
[41] "lifted_log_oPN9_oBt_comma_r_cB_cP_L226[12, 1]"
[42] "N7[12, 1]"
[43] "log.C1.Nf.3_7.m[11]"
[44] "log.C1.SPM.m[11]"
[45] "log.C1.nm.LbNf.m[11]"
[46] "C8.NAC.3.tot[12]"
[47] "C8.NAC.5.tot[12]"
[48] "mu.Gld[11, 1]"
[49] "mu.Gld[11, 2]"
[50] "mu.Gld[11, 3]"
[51] "mu.Gld[11, 4]"
[52] "mu.Gld[11, 5]"
[53] "mu.Gld[11, 6]"
[54] "mu.Gld[11, 7]"
[55] "mu.Gld[11, 8]"
[56] "mu.Gld[11, 9]"
[57] "mu.Gld[11, 10]"
[58] "mu.Gld[11, 11]"
[59] "mu.Gld[11, 12]"
[60] "mu.Gld[11, 13]"
[61] "mu.Gld[11, 14]"
[62] "mu.Gld[11, 15]"
[63] "mu.Gld[11, 16]"
[64] "mu.Gld[11, 17]"
[65] "mu.Gld[11, 18]"
[66] "mu.Gld[11, 19]"
[67] "mu.Gld[11, 20]"
[68] "mu.Gld[11, 21]"
[69] "mu.Gld[11, 22]"
[70] "mu.Gld[11, 23]"
[71] "mu.Gld[11, 24]"
[72] "lifted_log_oPC8_dot_2_dot_tot_oBt_cB_cP_L238[11]"
[73] "N2[11, 1]"
[74] "lifted_log_oPC8_dot_NAC_dot_4_dot_lab_oBt_cB_cP_L255[12]"
[75] "lifted_log_oPChw_dot_2SW_oBt_comma_r_cB_cP_L232[12, 1]"
[76] "log.R2SW.m[12, 1]"
[77] "N1[12, 1]"
[78] "lifted_log_oPC8_dot_NAC_dot_3_dot_tot_oBt_cB_cP_L252[12]"
[79] "lifted_log_oPC8_dot_NAC_dot_5_dot_tot_oBt_cB_cP_L257[12]"
[80] "prop_Gld[11, 1:24]"
[81] "log.CG2.m[11]"
[82] "Surv.eggs[11, 1]"
[83] "log.C2.tot.Lb.m[12]"
[84] "log.hwC2SW.m[12, 1]"
[85] "log.N2.m[12, 1]"
[86] "N10[13, 1]"
[87] "log.C2.Nf.3_7.m[12]"
[88] "log.C2.SPM.m[12]"
[89] "N2[12, 1]"
[90] "Surv.eggs[12, 1]"
[91] "N1[13, 1]"
[92] "log.N2.m[13, 1]"
[93] "N2[13, 1]"
[94] "Surv.eggs[13, 1]"

Run time

Run time in hours

	V_simple	V_det
Base	14.7	98.9

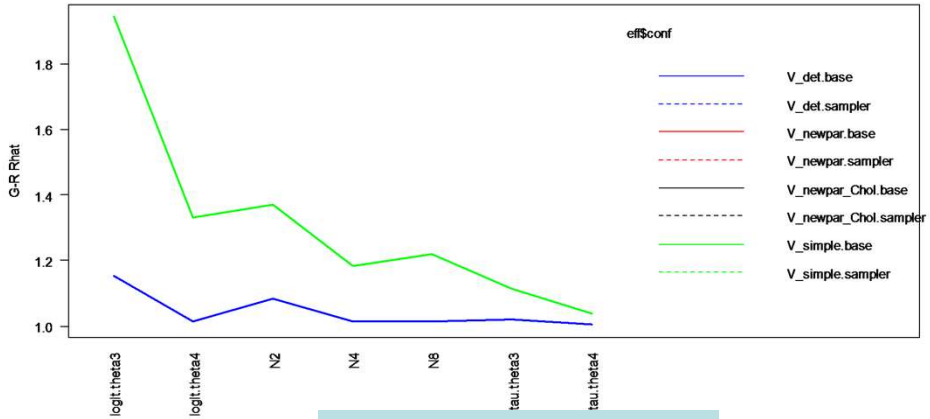
Relative to the fastest

	V_simple	V_det
Base	1	6.7

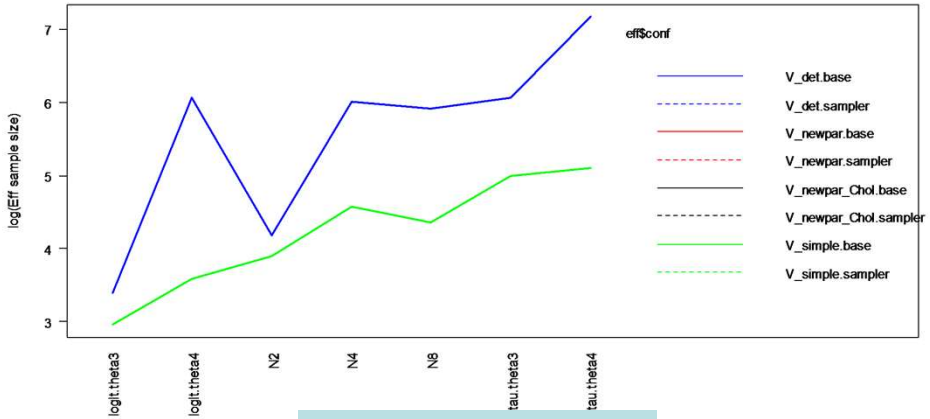
MCMC efficiency

➔ Improves algorithmic efficiency (mixing) but not necessarily computational efficiency

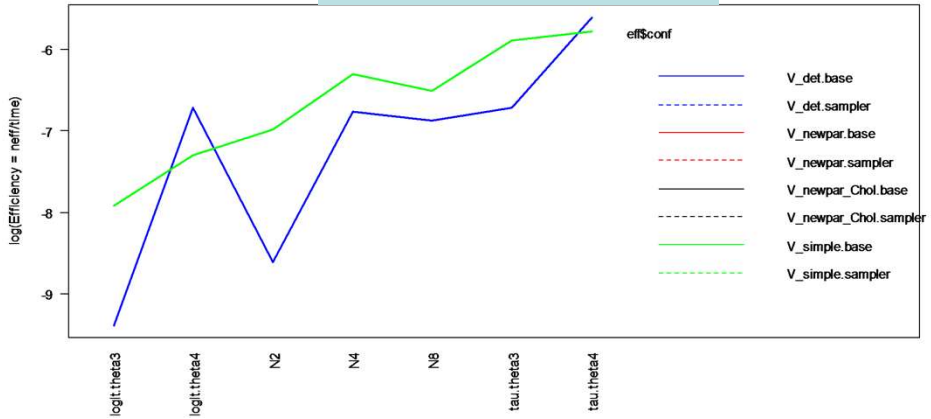
GR R_{hat}



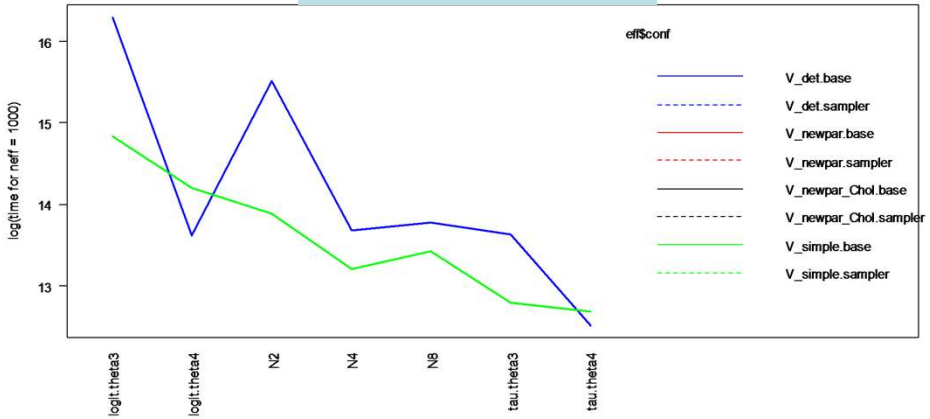
log(ESS)



log(ESS/run time)



log($time_{ESS=1000}$)



Comparing different configurations

- Benchmarking
- Baseline version (with good inits)
- Improve parameterization
  - Use deterministic transitions
  - Use customized distributions to integrate out transitions
  - Optimize prior for variance-covariance matrix
- Change default samplers

Customized distribution *V_newpar*

Baseline (deterministic)

$\theta_{t,1:Z}$ is a multivariate random walk (logit scale)

$$logit(\theta_{t,1:Z}) \sim MVNormal(logit(\theta_{t-1,1:Z}), \sum)$$

Conditionally on survival rate $\theta_{t,1:Z}$, the transition $N_{t,1:Z} \rightarrow N_{t+1,1:Z}$ is deterministic. In the log scale:

$$log(N_{t+1,1:Z}) = log(\theta_{t,1:Z}) + log(N_{t,1:Z})$$

Customized

➔ Build a customized sampling distribution that integrates two steps

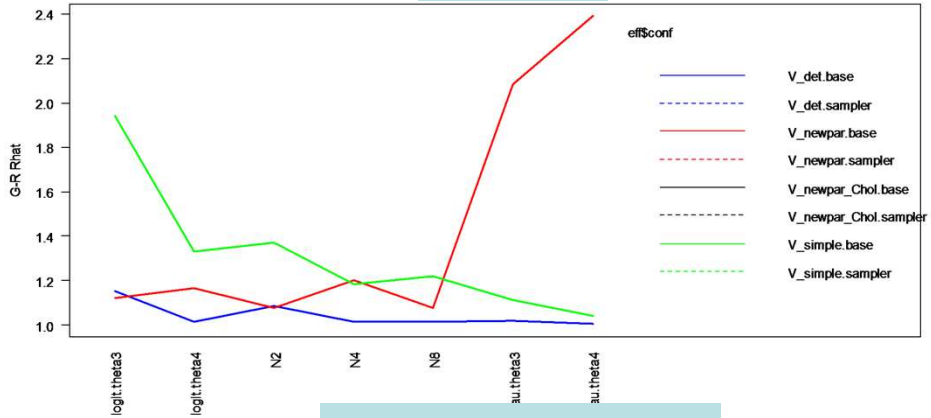
➔ to directly sample $log(N_{t+1,1:Z})$ in its pdf :

$$log(N_{t+1,1:Z}) \mid log(N_{t,1:Z}), logit(\theta_{t-1,1:Z}), \sum$$

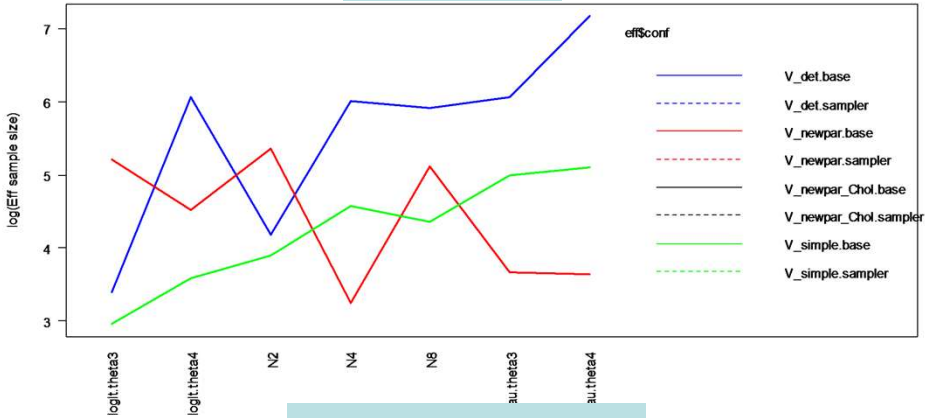
MCMC efficiency

➔ Improves mixing (algorithmic efficiency) for some but NOT all nodes (very low mixing for var-covar)

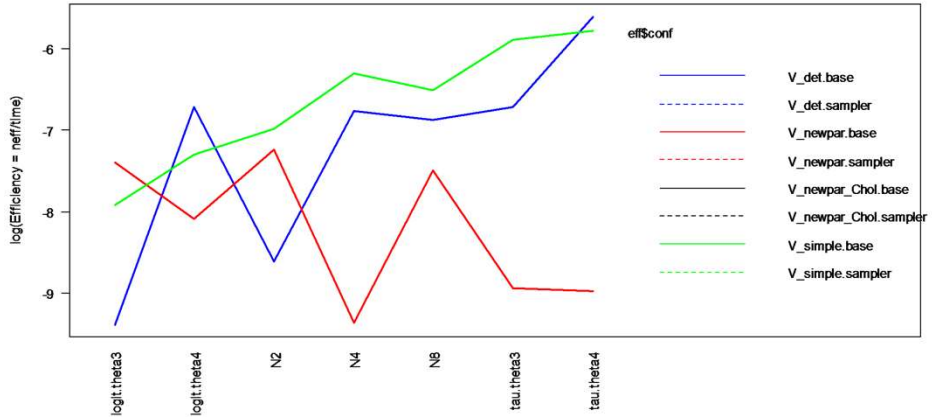
GR R_{hat}



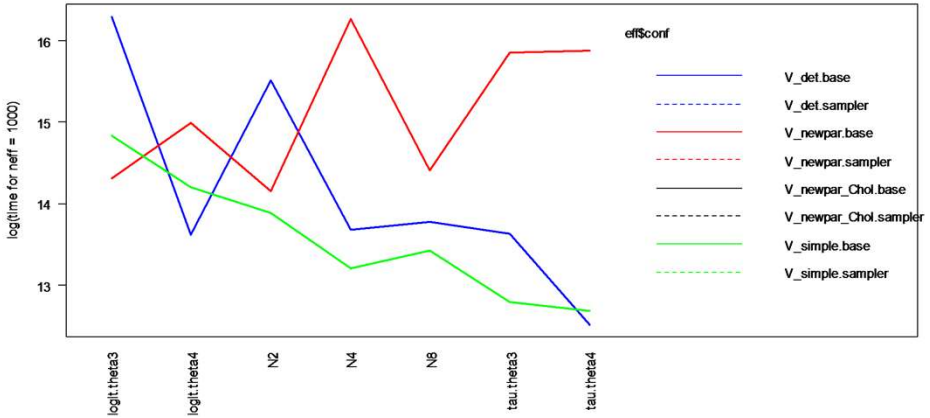
log(ESS)



log(ESS/run time)



log($time_{ESS=1000}$)



Comparing different configurations

- Benchmarking
- Baseline version (with good inits)
- Improve parameterization
  - Use deterministic transitions
  - Use customized distributions to integrate out transitions
  - Optimize prior for variance-covariance matrix
- Change default samplers

Default - block sampling

NIMBLE automatically sets up an Adaptive RW Metropolis Block Sampler (dim = 24) for the multivariate nodes

$logit(\theta_{t,1:Z})$ (*V_simple and V_det*)

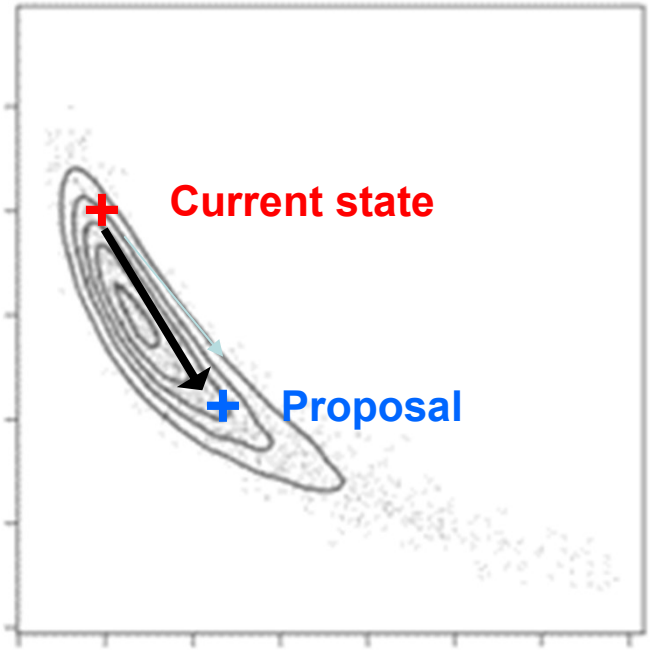
$logit(N_{t,1:Z})$ (*V_newpar*)

RW Metropolis Block sampling

→ A tradeoff between the gain of block sampling to explore joint posterior distribution and the loss of efficiency if the dimension of the joint posterior distribution is too high.

. May be more efficient to explore correlated joint distribution because of its capacity to propose a candidate that accounts for the covariance of mult var. nodes.

. But efficiency depends on the capacity to tune the var-covar matrix for the proposal. May be low if the dimension of the block is high. This inefficiency scales with the dimension of block sampler.



No block sampling *V_sampler*

➔ **Force simple scalar Adaptive RW Metropolis Sampler for all scalar components of nodes $logit(\theta_{t,1:Z})$ or $logit(N_{t,1:Z})$.**

NIMBLE

```
# define target nodes

# remove sampler for the target nodes
my.configureMCMC$removeSamplers(target)

# force sampler for the target node to be simple RW Metropolis-H
my.configureMCMC$addSampler(target, type = 'RW')
```

Run time

➔ Only slightly increases computational requirements

Run time in hours

	V_simple	V_det	V_newpar
Base	14.7	98.9	83.0
Sampler	18.8	107.6	99.5

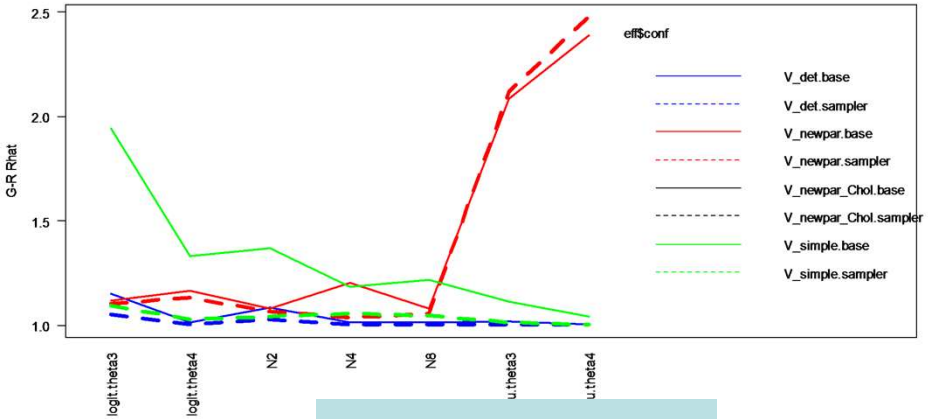
Relative to the fastest

	V_simple	V_det	V_newpar
Base	1	6.7	5.6
Sampler	1.3	7.3	6.8

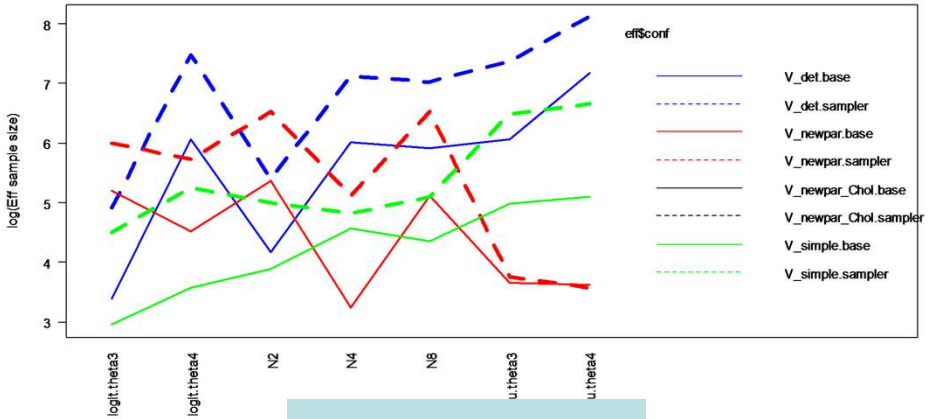
MCMC efficiency

➔ Improves overall computational efficiency for all configurations

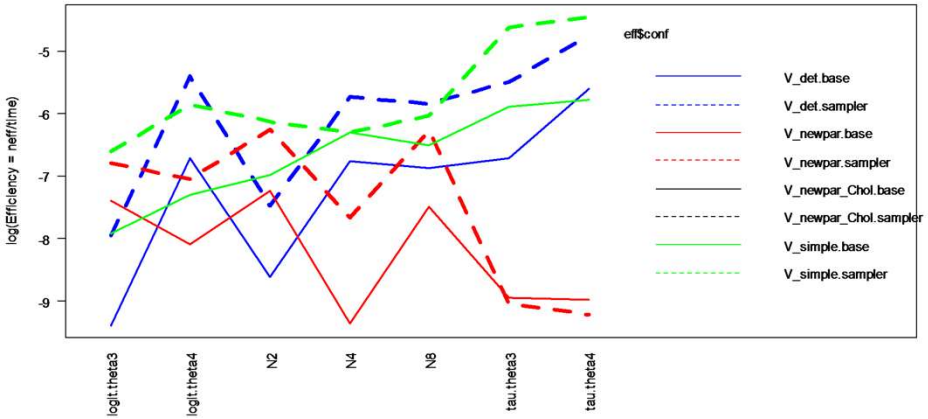
GR R_{hat}



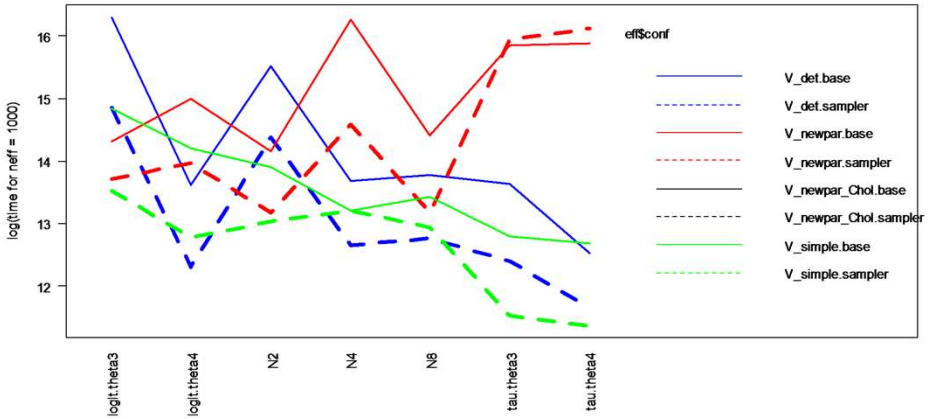
log(ESS)



log(ESS/run time)



log($time_{ESS=1000}$)



Conclusions and perspectives

Conclusions

- Default parameterization or sampler choices can be inefficient
- Substantive improvement can be obtained through careful model parameterization x customization of MCMC sampler

  NIMBLE is + +
- To be efficient, require a good understanding of model X sampler interactions

- Overall, in the example, efficiency remains low !
- Work under progress !

  Try different sampling strategy (STAN)