Avoiding tipping points in the management of ecological systems: a non-parametric Bayesian approach

Carl Boettiger[a,*], Marc Mangel[a], Stephan B. Munch[b]

[a] *Center for Stock Assessment Research, Department of Applied Math and Statistics, University of California, Mail Stop SOE-2, Santa Cruz, CA 95064, USA*

[b] *Southwest Fisheries Science Center, National Oceanic and Atmospheric Administration, 110 Shaffer Road, Santa Cruz, CA 95060, USA*

```
options(xtable.print.comment = FALSE)
options(xtable.type = "latex", table.placement = "H")
opts_knit$set(progress = TRUE, verbose = TRUE)
opts_chunk$set(dev = c("pdf", "png"), fig.width = 5.5, fig.height = 4, cache.path = "cache/nonparametric-l
    cache = TRUE, external = TRUE)
opts_chunk$set(warning = FALSE, message = FALSE, comment = NA, tidy = FALSE)
toggle = "markup"
theme_set(theme_bw(base_size = 12))

require(modeest)
posterior.mode <- function(x) {
  mlv(x, method="shorth")$M
}

f <- RickerAllee
p <- c(2, 8, 5)
K <- 10  # approx, a li'l' less
allee <- 5 # approx, a li'l' less

sigma_g <- 0.05
sigma_m <- 0.0
z_g <- function() rlnorm(1, 0, sigma_g)
z_m <- function() 1
x_grid <- seq(0, 1.5 * K, length=50)
```

*Corresponding author
Email address: cboettig@ucsc.edu (Carl Boettiger)

June 24, 2013

```r
h_grid <- x_grid
profit <- function(x,h) pmin(x, h)
delta <- 0.01
OptTime <- 50  # stationarity with unstable models is tricky thing
reward <- 0
xT <- 0
Xo <-  allee+.5# observations start from
x0 <- K # simulation under policy starts from
Tobs <- 40
MaxT = 1000 # timeout for value iteration convergence

  set.seed(1234)
  #harvest <- sort(rep(seq(0, .5, length=7), 5))
  x <- numeric(Tobs)
  x[1] <- Xo
  nz <- 1
  for(t in 1:(Tobs-1))
    x[t+1] = z_g() * f(x[t], h=0, p=p)
  obs <- data.frame(x = c(rep(0,nz),
                          pmax(rep(0,Tobs-1), x[1:(Tobs-1)])),
                    y = c(rep(0,nz),
                          x[2:Tobs]))
raw_plot <- ggplot(data.frame(time = 1:Tobs, x=x), aes(time,x)) + geom_line()
raw_plot


set.seed(12345)
estf <- function(p){
    mu <- f(obs$x,0,p)
    -sum(dlnorm(obs$y, log(mu), p[4]), log=TRUE)
}
par <- c(p[1]*rlnorm(1,0,.1),
         p[2]*rlnorm(1,0,.1),
         p[3]*rlnorm(1,0, .1),
         sigma_g * rlnorm(1,0,.1))
o <- optim(par, estf, method="L", lower=c(1e-5,1e-5,1e-5,1e-5))
f_alt <- f
```

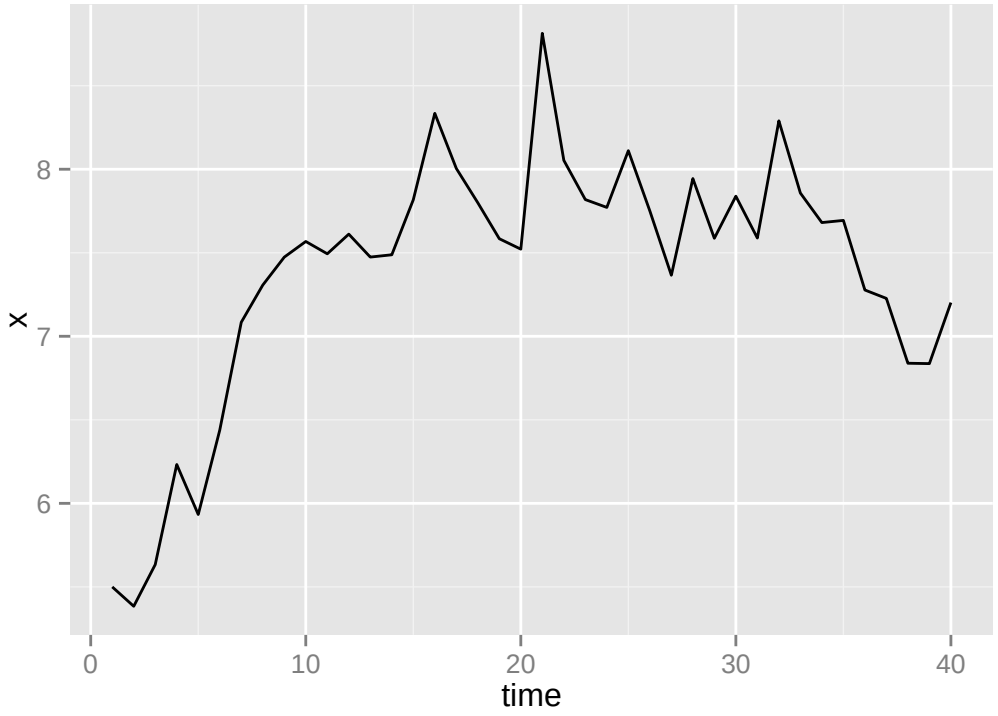

Figure 1: plot of chunk obs

```
p_alt <- c(as.numeric(o$par[1]), as.numeric(o$par[2]), as.numeric(o$par[3]))
sigma_g_alt <- as.numeric(o$par[4])

est <- list(f = f_alt, p = p_alt, sigma_g = sigma_g_alt, mloglik=o$value)

true_means <- sapply(x_grid, f, 0, p)
est_means <- sapply(x_grid, est$f, 0, est$p)

#inv gamma has mean b / (a - 1) (assuming a>1) and variance b ^ 2 / ((a - 2) * (a - 1) ^ 2) (assuming a>2
s2.p <- c(5,5)
d.p = c(10, 1/0.1)

gp <- gp_mcmc(obs$x, y=obs$y, n=1e5, s2.p = s2.p, d.p = d.p)
gp_dat <- gp_predict(gp, x_grid, burnin=1e4, thin=300)

gp_assessment_plots <- summary_gp_mcmc(gp, burnin=1e4, thin=300)

# Summarize the GP model
tgp_dat <-
```

```
    data.frame(  x = x_grid,
                 y = gp_dat$E_Ef,
                 ymin = gp_dat$E_Ef - 2 * sqrt(gp_dat$E_Vf),
                 ymax = gp_dat$E_Ef + 2 * sqrt(gp_dat$E_Vf) )

y <- x
N <- length(x);
jags.data <- list("N","y")
n.chains <- 6
n.iter <- 1e6
n.burnin <- floor(10000)
n.thin <- max(1, floor(n.chains * (n.iter - n.burnin)/1000))
n.update <- 10

stdQ_prior_p <- c(1e-6, 100)
stdR_prior_p <- c(1e-6, .1)
stdQ_prior  <- function(x) dunif(x, stdQ_prior_p[1], stdQ_prior_p[2])
stdR_prior  <- function(x) dunif(x, stdR_prior_p[1], stdR_prior_p[2])

K_prior_p <- c(0.01, 20.0)
r0_prior_p <- c(0.01, 6.0)
theta_prior_p <- c(0.01, 20.0)

bugs.model <-
paste(sprintf(
"model{
  K     ~ dunif(%s, %s)
  r0    ~ dunif(%s, %s)
  theta ~ dunif(%s, %s)
  stdQ ~ dunif(%s, %s)",
  K_prior_p[1], K_prior_p[2],
  r0_prior_p[1], r0_prior_p[2],
  theta_prior_p[1], theta_prior_p[2],
  stdQ_prior_p[1], stdQ_prior_p[2]),

  "
  iQ <- 1 / (stdQ * stdQ);
```

```
  y[1] ~ dunif(0, 10)
  for(t in 1:(N-1)){
    mu[t] <- log(y[t]) + r0 * (1 - y[t]/K)* (y[t] - theta) / K
    y[t+1] ~ dlnorm(mu[t], iQ)
  }
}")
writeLines(bugs.model, "allen_process.bugs")

K_prior     <- function(x) dunif(x, K_prior_p[1], K_prior_p[2])
r0_prior <- function(x) dunif(x, r0_prior_p[1], r0_prior_p[2])
theta_prior <- function(x) dunif(x, theta_prior_p[1], theta_prior_p[2])
par_priors  <- list(K = K_prior, deviance = function(x) 0 * x,
                    r0 = r0_prior, theta = theta_prior,
                    stdQ = stdQ_prior)

jags.params=c("K","r0","theta","stdQ") # be sensible about the order here
jags.inits <- function(){
  list("K"= 10 * rlnorm(1,0, 0.1),
       "r0"= 1 * rlnorm(1,0, 0.1) ,
       "theta"=   5 * rlnorm(1,0, 0.1) ,
       "stdQ"= abs( 0.1 * rlnorm(1,0, 0.1)),
       .RNG.name="base::Wichmann-Hill", .RNG.seed=123)
}

set.seed(1234)
# parallel refuses to take variables as arguments (e.g. n.iter = 1e5 works, but n.iter = n doesn't)
allen_jags <- do.call(jags.parallel, list(data=jags.data, inits=jags.inits,
                                          jags.params, n.chains=n.chains,
                                          n.iter=n.iter, n.thin=n.thin,
                                          n.burnin=n.burnin,
                                          model.file="allen_process.bugs"))

# Run again iteratively if we haven't met the Gelman-Rubin convergence criterion
recompile(allen_jags) # required for parallel

Compiling model graph
   Resolving undeclared variables
```

```
   Allocating nodes
   Graph Size: 328

Initializing model

Compiling model graph
   Resolving undeclared variables
   Allocating nodes
   Graph Size: 328

Initializing model

Compiling model graph
   Resolving undeclared variables
   Allocating nodes
   Graph Size: 328

Initializing model

Compiling model graph
   Resolving undeclared variables
   Allocating nodes
   Graph Size: 328

Initializing model

Compiling model graph
   Resolving undeclared variables
   Allocating nodes
   Graph Size: 328

Initializing model

Compiling model graph
   Resolving undeclared variables
   Allocating nodes
```

```
   Graph Size: 328

Initializing model

allen_jags <- do.call(autojags,
                                          list(object=allen_jags, n.update=n.update,
                          n.iter=n.iter, n.thin = n.thin))

tmp <- lapply(as.mcmc(allen_jags), as.matrix) # strip classes the hard way...
allen_posteriors <- melt(tmp, id = colnames(tmp[[1]]))
names(allen_posteriors) = c("index", "variable", "value", "chain")
plot_allen_traces <- ggplot(allen_posteriors) + geom_line(aes(index, value)) +
  facet_wrap(~ variable, scale="free", ncol=1)

allen_priors <- ddply(allen_posteriors, "variable", function(dd){
    grid <- seq(min(dd$value), max(dd$value), length = 100)
    data.frame(value = grid, density = par_priors[[dd$variable[1]]](grid))
})
plot_allen_posteriors <- ggplot(allen_posteriors, aes(value)) +
  stat_density(geom="path", position="identity", alpha=0.7) +
  geom_line(data=allen_priors, aes(x=value, y=density), col="red") +
  facet_wrap(~ variable, scale="free", ncol=3)

A <- allen_posteriors
A$index <- A$index + A$chain * max(A$index) # Combine samples across chains by renumbering index
pardist <- acast(A, index ~ variable)
bayes_coef <- apply(pardist,2, posterior.mode)
bayes_pars <- unname(c(bayes_coef["r0"], bayes_coef["K"], bayes_coef["theta"])) # parameters formatted fo
allen_f <- function(x,h,p) unname(RickerAllee(x,h, unname(p[c("r0", "K", "theta")])))
allen_means <- sapply(x_grid, f, 0, bayes_pars)
bayes_pars

[1] 0.5003 7.7115 4.6039

head(pardist)

        K deviance     r0    stdQ theta
```

```
170 7.898    34.41 0.6668 0.05566 3.956
171 7.745    32.76 0.7554 0.05137 4.668
172 7.577    31.66 1.0705 0.04562 4.073
173 7.859    31.26 1.5647 0.04238 5.087
174 7.821    30.67 1.7348 0.04662 5.230
175 8.003    34.90 1.4495 0.05005 5.444
```

```
K_prior_p <- c(0.01, 40.0)
r0_prior_p <- c(0.01, 20.0)
bugs.model <-
paste(sprintf(
"model{
  K    ~ dunif(%s, %s)
  r0    ~ dunif(%s, %s)
  stdQ ~ dunif(%s, %s)",
  K_prior_p[1], K_prior_p[2],
  r0_prior_p[1], r0_prior_p[2],
  stdQ_prior_p[1], stdQ_prior_p[2]),
  "
  iQ <- 1 / (stdQ * stdQ);
  y[1] ~ dunif(0, 10)
  for(t in 1:(N-1)){
    mu[t] <- log(y[t]) + r0 * (1 - y[t]/K)
    y[t+1] ~ dlnorm(mu[t], iQ)
  }
}")
writeLines(bugs.model, "ricker_process.bugs")

K_prior     <- function(x) dunif(x, K_prior_p[1], K_prior_p[2])
r0_prior <- function(x) dunif(x, r0_prior_p[1], r0_prior_p[2])
par_priors <- list(K = K_prior, deviance = function(x) 0 * x,
                   r0 = r0_prior, stdQ = stdQ_prior)

jags.params=c("K","r0", "stdQ")
jags.inits <- function(){
  list("K"= 10 * rlnorm(1,0,.5),
       "r0"= rlnorm(1,0,.5),
```

```
       "stdQ"=sqrt(0.05) * rlnorm(1,0,.5),
       .RNG.name="base::Wichmann-Hill", .RNG.seed=123)
}
set.seed(12345)
ricker_jags <- do.call(jags.parallel,
                       list(data=jags.data, inits=jags.inits,
                            jags.params, n.chains=n.chains,
                            n.iter=n.iter, n.thin=n.thin, n.burnin=n.burnin,
                            model.file="ricker_process.bugs"))
recompile(ricker_jags)
```

```
Compiling model graph
   Resolving undeclared variables
   Allocating nodes
   Graph Size: 249

Initializing model

Compiling model graph
   Resolving undeclared variables
   Allocating nodes
   Graph Size: 249

Initializing model

Compiling model graph
   Resolving undeclared variables
   Allocating nodes
   Graph Size: 249

Initializing model

Compiling model graph
   Resolving undeclared variables
   Allocating nodes
   Graph Size: 249
```

```
Initializing model

Compiling model graph
   Resolving undeclared variables
   Allocating nodes
   Graph Size: 249

Initializing model

Compiling model graph
   Resolving undeclared variables
   Allocating nodes
   Graph Size: 249

Initializing model
```

```
ricker_jags <- do.call(autojags,
                       list(object=ricker_jags, n.update=n.update,
                                                n.iter=n.iter, n.thin = n.thin,
                                                progress.bar="none"))

tmp <- lapply(as.mcmc(ricker_jags), as.matrix) # strip classes the hard way...
ricker_posteriors <- melt(tmp, id = colnames(tmp[[1]]))
names(ricker_posteriors) = c("index", "variable", "value", "chain")
plot_ricker_traces <- ggplot(ricker_posteriors) + geom_line(aes(index, value)) +
  facet_wrap(~ variable, scale="free", ncol=1)

ricker_priors <- ddply(ricker_posteriors, "variable", function(dd){
    grid <- seq(min(dd$value), max(dd$value), length = 100)
    data.frame(value = grid, density = par_priors[[dd$variable[1]]](grid))
})
# plot posterior distributions
plot_ricker_posteriors <- ggplot(ricker_posteriors, aes(value)) +
  stat_density(geom="path", position="identity", alpha=0.7) +
  geom_line(data=ricker_priors, aes(x=value, y=density), col="red") +
  facet_wrap(~ variable, scale="free", ncol=2)
```

```
A <- ricker_posteriors
A$index <- A$index + A$chain * max(A$index) # Combine samples across chains by renumbering index
ricker_pardist <- acast(A, index ~ variable)
bayes_coef <- apply(ricker_pardist,2, posterior.mode)
ricker_bayes_pars <- unname(c(bayes_coef["r0"], bayes_coef["K"]))
ricker_f <- function(x,h,p){
  sapply(x, function(x){
    x <- pmax(0, x-h)
    pmax(0, x * exp(p["r0"] * (1 - x / p["K"] )) )
  })
}
ricker_means <- sapply(x_grid, Ricker, 0, ricker_bayes_pars[c(1,2)])
head(ricker_pardist)
```

```
         K deviance      r0    stdQ
170  7.727    35.73 0.29440 0.05197
171  7.949    37.93 0.18845 0.05872
172 32.183    44.54 0.02474 0.05688
173  8.149    37.59 0.12204 0.05407
174  7.460    39.01 0.38643 0.05495
175  7.361    35.54 0.23805 0.05150
```

```
ricker_bayes_pars
```

```
[1] 0.1997 7.5976
```

```
r0_prior_p <- c(.0001, 10.0)
theta_prior_p <- c(.0001, 10.0)
K_prior_p <- c(.0001, 40.0)
bugs.model <-
paste(sprintf(
"model{
  r0    ~ dunif(%s, %s)
  theta    ~ dunif(%s, %s)
  K    ~ dunif(%s, %s)
  stdQ ~ dunif(%s, %s)",
  r0_prior_p[1], r0_prior_p[2],
```

```
  theta_prior_p[1], theta_prior_p[2],
  K_prior_p[1], K_prior_p[2],
  stdQ_prior_p[1], stdQ_prior_p[2]),

  "
  iQ <- 1 / (stdQ * stdQ);

  y[1] ~ dunif(0, 10)
  for(t in 1:(N-1)){
    mu[t] <- log(r0)  + theta * log(y[t]) - log(1 + pow(abs(y[t]), theta) / K)
    y[t+1] ~ dlnorm(mu[t], iQ)
  }
}")
writeLines(bugs.model, "myers_process.bugs")

K_prior     <- function(x) dunif(x, K_prior_p[1], K_prior_p[2])
r_prior     <- function(x) dunif(x, r0_prior_p[1], r0_prior_p[2])
theta_prior <- function(x) dunif(x, theta_prior_p[1], theta_prior_p[2])
par_priors <- list( deviance = function(x) 0 * x, K = K_prior,
                    r0 = r_prior, theta = theta_prior,
                    stdQ = stdQ_prior)

jags.params=c("r0", "theta", "K", "stdQ")
jags.inits <- function(){
  list("r0"= 1 * rlnorm(1,0,.1),
       "K"=    10 * rlnorm(1,0,.1),
       "theta" = 1 * rlnorm(1,0,.1),
       "stdQ"= sqrt(0.2) * rlnorm(1,0,.1),
       .RNG.name="base::Wichmann-Hill", .RNG.seed=123)
}
set.seed(12345)
myers_jags <- do.call(jags.parallel,
                      list(data=jags.data, inits=jags.inits,
                                              jags.params, n.chains=n.chains,
                                              n.iter=n.iter, n.thin=n.thin,
                           n.burnin=n.burnin,
                           model.file="myers_process.bugs"))
```

```
recompile(myers_jags)
```

```
Compiling model graph
   Resolving undeclared variables
   Allocating nodes
   Graph Size: 406

Initializing model

Compiling model graph
   Resolving undeclared variables
   Allocating nodes
   Graph Size: 406

Initializing model

Compiling model graph
   Resolving undeclared variables
   Allocating nodes
   Graph Size: 406

Initializing model

Compiling model graph
   Resolving undeclared variables
   Allocating nodes
   Graph Size: 406

Initializing model

Compiling model graph
   Resolving undeclared variables
   Allocating nodes
   Graph Size: 406

Initializing model
```

```
Compiling model graph
   Resolving undeclared variables
   Allocating nodes
   Graph Size: 406

Initializing model
```

```
myers_jags <- do.call(autojags,
                      list(myers_jags, n.update=n.update,
                           n.iter=n.iter, n.thin = n.thin,
                           progress.bar="none"))
```

```
tmp <- lapply(as.mcmc(myers_jags), as.matrix) # strip classes
myers_posteriors <- melt(tmp, id = colnames(tmp[[1]]))
names(myers_posteriors) = c("index", "variable", "value", "chain")
plot_myers_traces <- ggplot(myers_posteriors) + geom_line(aes(index, value)) +
  facet_wrap(~ variable, scale="free", ncol=1)

par_prior_curves <- ddply(myers_posteriors, "variable", function(dd){
    grid <- seq(min(dd$value), max(dd$value), length = 100)
    data.frame(value = grid, density = par_priors[[dd$variable[1]]](grid))
})
plot_myers_posteriors <- ggplot(myers_posteriors, aes(value)) +
  stat_density(geom="path", position="identity", alpha=0.7) +
  geom_line(data=par_prior_curves, aes(x=value, y=density), col="red") +
  facet_wrap(~ variable, scale="free", ncol=3)

A <- myers_posteriors
A$index <- A$index + A$chain * max(A$index) # Combine samples across chains by renumbering index
myers_pardist <- acast(A, index ~ variable)
bayes_coef <- apply(myers_pardist,2, posterior.mode) # much better estimates
myers_bayes_pars <- unname(c(bayes_coef["r0"], bayes_coef["theta"], bayes_coef["K"]))
myers_means <- sapply(x_grid, Myer_harvest, 0, myers_bayes_pars)
myers_f <- function(x,h,p) Myer_harvest(x, h, p[c("r0", "theta", "K")])
head(myers_pardist)
```

```
        K deviance     r0    stdQ  theta
```

```
170 20.51    37.16 1.3995 0.04136 0.9864
171 17.10    41.09 0.6076 0.05170 1.8684
172 24.40    36.33 1.1692 0.04307 1.0715
173 27.56    37.46 0.9533 0.05550 1.2000
174 22.21    34.94 0.5140 0.04203 1.8694
175 36.05    33.49 0.3089 0.03977 2.1519
```

```
myers_bayes_pars
```

```
[1]  0.5129  1.0894 23.9914
```

```
models <- data.frame(x=x_grid,
                                        GP=tgp_dat$y,
                                        True=true_means,
                     MLE=est_means,
                                        Ricker=ricker_means,
                     Allen = allen_means,
                     Myers = myers_means)
models <- melt(models, id="x")

# some labels
names(models) <- c("x", "method", "value")

# labels for the colorkey too
model_names = c("GP", "True", "MLE", "Ricker", "Allen", "Myers")
colorkey=cbPalette
names(colorkey) = model_names

# uses expected values from GP, instead of integrating over posterior
#matrices_gp <- gp_transition_matrix(gp_dat$E_Ef, gp_dat$E_Vf, x_grid, h_grid)
matrices_gp <- gp_transition_matrix(gp_dat$Ef_posterior, gp_dat$Vf_posterior, x_grid, h_grid)
opt_gp <- value_iteration(matrices_gp, x_grid, h_grid, MaxT, xT, profit, delta, reward)

matrices_true <- f_transition_matrix(f, p, x_grid, h_grid, sigma_g)
opt_true <- value_iteration(matrices_true, x_grid, h_grid, OptTime=MaxT, xT, profit, delta=delta)
matrices_estimated <- f_transition_matrix(est$f, est$p, x_grid, h_grid, est$sigma_g)
opt_estimated <- value_iteration(matrices_estimated, x_grid, h_grid, OptTime=MaxT, xT, profit, delta=delta
```

```
matrices_allen <- parameter_uncertainty_SDP(allen_f, x_grid, h_grid, pardist, 4)
opt_allen <- value_iteration(matrices_allen, x_grid, h_grid, OptTime=MaxT, xT, profit, delta=delta)

matrices_ricker <- parameter_uncertainty_SDP(ricker_f, x_grid, h_grid, as.matrix(ricker_pardist), 3)
opt_ricker <- value_iteration(matrices_ricker, x_grid, h_grid, OptTime=MaxT, xT, profit, delta=delta)

matrices_myers <- parameter_uncertainty_SDP(myers_f, x_grid, h_grid, as.matrix(myers_pardist), 4)
myers_alt <- value_iteration(matrices_myers, x_grid, h_grid, OptTime=MaxT, xT, profit, delta=delta)

OPT = data.frame(GP = opt_gp$D, True = opt_true$D, MLE = opt_estimated$D, Ricker = opt_ricker$D, Allen = o
colorkey=cbPalette
names(colorkey) = names(OPT)

sims <- lapply(OPT, function(D){
  set.seed(1)
  lapply(1:100, function(i)
    ForwardSimulate(f, p, x_grid, h_grid, x0, D, z_g, profit=profit, OptTime=OptTime)
  )
})

dat <- melt(sims, id=names(sims[[1]][[1]]))
sims_data <- data.table(dat)
setnames(sims_data, c("L1", "L2"), c("method", "reps"))
# Legend in original ordering please, not alphabetical:
sims_data$method = factor(sims_data$method, ordered=TRUE, levels=names(OPT))

Profit <- sims_data[, sum(profit), by=c("reps", "method")]
tmp <- dcast(Profit, reps ~ method)
#tmp$Allen <- tmp[,"Allen"] + rnorm(dim(tmp)[1], 0, 1) # jitter for plotting
tmp <- tmp / tmp[,"True"]
tmp <- melt(tmp[2:dim(tmp)[2]])
actual_over_optimal <-subset(tmp, variable != "True")

allen_deviance <- -2*posterior.mode(pardist[,'deviance'])
ricker_deviance <- -2*posterior.mode(ricker_pardist[,'deviance'])
myers_deviance <- -2*posterior.mode(myers_pardist[,'deviance'])
true_deviance <- 2*estf(c(p, sigma_g))
```

```
mle_deviance <- 2*estf(c(est$p, est$sigma_g))
xtable::xtable(as.table(c(Allen = allen_deviance, Ricker=ricker_deviance, Myers=myers_deviance, True=true_
```

```
% latex table generated in R 3.0.1 by xtable 1.7-1 package
% Thu Jun 20 16:29:00 2013
\begin{table}[ht]
\centering
\begin{tabular}{rr}
  \hline
 & x \\
  \hline
Allen & -65.85 \\
  Ricker & -71.85 \\
  Myers & -70.92 \\
  True & -61.08 \\
  MLE & -889.99 \\
   \hline
\end{tabular}
\end{table}
```


Abstract

Decision-theoretic methods often rely on simple parametric models of ecological dynamics to compare the value of a potential sequence of actions. Unfortunately, such simple models rarely capture the complexity or uncertainty found in most real ecosystems.

We demonstrate how nonparametric Bayesian models can provide robust, nearly-optimal solutions to decision making under uncertainty when *we don't know the correct model* to use.

While methods that account for *parametric* uncertainty can be very successful with the right model, structural uncertainty of not knowing what model best approximates the dynamics poses considerably greater difficulty.

Non-parametric Bayesian methods offer a promising statistical approach for predictive modeling of ecological dynamics in regions of state space where the data is adequate, while at the same time offering more flexible patterns with greater uncertainty outside the observed data. This contrasts from simple parametric models which provide relatively constant level of uncertainty in regions with and without adequate data. The consequence of such misplaced confidence outside the data can lead to highly undesirable results that may be avoided with the more flexible non-parametric Bayesian approach.

Introduction

Decision making under uncertainty is a ubiquitous challenge of natural resource management and conservation.

Here we illustrate how a stochastic dynamic programming algorithm can be driven by the predictions from a Gaussian process model, sidestepping the need for an accurate model-based description of the sustem dynamics.

Decision-theoretic (or optimal control) tools require a model that can assign probabilities of future states (e.g. stock size of a fishery) given the current state and a proposed action (e.g. fishing harvest or effort). The decision maker then seeks to determining the course of actions (also referred to as the policy) that maximizes the expected value of some objective function (such as net present value derived from the resource over time. (The approach can be adapted to permit alternatives to maximizing expectation, such as minimizing the maximum cost or damage that might be incurred. See @Polasky2011). Management frequently faces a sequential decision-making problem – after selecting an action, the decision-maker may receive new information about the current state and must again choose an appropriate action – i.e. setting the harvest limits each year based on stock assessments the year prior.

Traditional approaches to optimal control (Pontryagin's principle, stochastic dynamic programming) rely on knowledge of the state equation. Such simple parametric models, frequently justified by mechanistic underpinnings are used to provide probabilities of future states. Fisheries literature, which enjoys some of the longest history in the use of the optimal control framework, provides a clear example. Models such as the Ricker or Beverton Holt curves are ubiquitous [too strong? references?] Parametric models can reflect variability introduced by a stochastic environment or demographic process my incorporating random shocks from a given distribution.

As the parameter values for these models must be estimated from limited data,
there will always be some uncertainty associated with these values. This uncertainty further compounds the intrinsic variability introduced by demographic or environmental noise. The degree of uncertainty in the parameter values can be inferred from the data and reflected in the estimates of the transition probabilities.

Unfortunately, estimates of parameter uncertainty are only as good as the parametric models themselves. Too often, we do not understand the system dynamics well enough to know if a model provides a good approximation over the relevant range of states and timescales (criteria that we loosely refer to as defining the "right" or "true" model.) So called structural or model uncertainty is a more difficult problem than parametric uncertainty. Typical solutions involve either model choice, or model averaging. Either approach remains limited by the degree to which any of the proposed models are sufficiently close to the right model.

The nature of decision-making problems provides a convenient way to compare models. Rather than compare models in terms of best fit to data or fret over the appropriate penalty for model complexity, model performance is defined in the concrete terms of the decision-maker's objective function, which we will take as given. (Much argument can be made over the 'correct' objective function, e.g. how to account for the social value of fish left in the sea vs. the commercial value of fish harvested; see @Halpern2013 for further discussion of this issue. Alternatively, we can always compare model performance across multiple potential objective functions.) The

decision-maker does not necessarily need a model that provides the best mechanistic understanding or the best long-term outcome, but rather the one that best estimates the probabilities of being in different states as a result of the possible actions.

While simple mechanistic models can nevertheless provide important insights into long-term outcomes, such approaches are not well-suited for use in forecasting outcomes of potential management options. Non-parametric approaches offer a more flexible alternative that can both more accurately reflect the data available while also representing greater uncertainty in areas (of state-space) where data is lacking.

We demonstrate how a Gaussian Process model of stock recruitment can lead to nearly optimal management through stochastic dynamic programming, comperable to knowing the correct structural equation for the underlying simulation. Meanwhile, parametric models that do not match the underlying dynamics can perform very poorly, even though they fit the data as well as the true model.

Ecological research and management strategy should pay closer attention to the opportunities and challenges nonparametric modeling can offer.

Non-parametric models can better represent uncertainties outside of observed data while also better capturing the dynamics in the region observed. Advances in the theory and computational implementations of nonparametric methods make them ripe for such applications. We use the classic problem of optimal harvest of a marine fishery to illustrate how the nonparametric approach of Gaussian processes can be applied. We will compare Bayesian implementations of both nonparametric and parametric models, which best allow us to capture the uncertainty of model estimation in either case and permits a more natural comparison between approaches.

Terminology: "Non-parametric"

The term non-parametric to describe modeling approaches such as Gaussian processes is a common source of confusion, notwithstanding the fact that the model is still specified by parameters. Some literature has introduced the term "semi-parametric" in tacit acknowledgment of this, which no doubt only contributes to the confusion. The problem is further exacerbated by the several meanings assigned to the term in statistics: (a) that the method does not assume some particular probability distribution (e.g. a non-parametric bootstrap) or (b) that the method does not assume a fixed structure to the model. Our use is that of (b), in contrast to the classical approaches that do. The difference is clearest by way of example.

Approach and Methods

Underlying Model

Concerns over the potential for tipping points in ecological dynamics (Scheffer et al. 2001) highlight the dangers of uncertainty in ecological management and pose a substantial challenge to existing decision-theoretic approaches

(Brozović and Schlenker 2011). Because intervention is often too late after a tipping point has been crossed (but see Hughes et al. (2013)), management is most often concerned with avoiding potentially catastrophic tipping points before any data is available at or following a transition that would more clearly reveal these regime shift dynamics (e.g. Bestelmeyer et al. 2012).

To illustrate the value of the non-parametric Bayesian approach to management, we focus on example of a system containing such a tipping point whose dynamics can still be described by a simple, one-dimensional parametric model.

We will focus on a simple parametric model for a single species (derived from fist principles by Allen et al. 2005) as our underlying "reality".

$$X_{t+1} = Z_t f(S_t)$$

$$S_t = X_t - h_t$$

$$f(S_t) = S_t e^{r\left(1-\frac{S_t}{K}\right)(S_t - C)}$$

Where Z_t is multiplicative noise function with mean 1, representing stochastic growth. We will consider log-normal noise with shape parameter σ_g. We start with an example in which the parameters are $r = 2$, $K = 8$, $C = 5$ and $\sigma_g = 0.1$.

As a low-dimensional system completely described by three parameters, this scenario should if anything be favorable to a parametric-based approach. This model contains an Allee effect, or tipping point, below which the population is not self-sustaining and shrinks to zero (Courchamp, Berec, and Gascoigne 2008).

Both parametric and nonparametric approaches will require training data on which to base their model of the process. We generate the training data under the model described in Eq 1 for $T_{\text{obs}} = 40$ time steps, under a known but not necessarily optimal sequence of harvest intensities, h_t. For simplicity we imagine a fishery that started from zero harvest pressure and has been gradually increasing the harvest.

Should we include any emprical examples?

The optimal control problem in fisheries management

In our example, we focus on the problem in which a manager must set the harvest level for a marine fishery each year to maximize the net present value of the resource, given an estimated stock size from the year before. The economic value and ecological concern have made marine fisheries the crucible for much of the founding work (Gordon 1954; Reed 1979; May et al. 1979; Ludwig and Walters 1982) in managing ecosystems under uncertainty. Global trends (Worm et al. 2006) and controversy (Hilborn 2007; Worm et al. 2009) have made understanding these challenges all the more pressing.

To permit comparisons against a theoretical optimum we will consider data on the stock dynamics simulated from a simple parametric model in which recruitment of the fish stock X_{t+1} in the following year is a stochastic process governed by a function f of the current stock X_t, selected harvest policy h_t, and noise process Z,

$$X_{t+1} = Z_t f(X_t, h_t)$$

Given parameters for the function f and probability distribution Z, along with a given economic model determining the price/profit $\Pi(X_t, h_t)$ realized in a given year given a choice of harvest h_t and observed stock X_t. This problem can be solved exactly for discretized values of stock X and policy h using stochastic dynamic programming (SDP) (Mangel 1985). Problems of this sort underpin much marine fisheries management today.

A crux of this approach is correctly specifying the functional form of f, along with its parameters. The standard approach uses one of a handful of common parametric models representing the stock-recruitment relationship, usually after estimating the model parameters from any available existing data. Uncertainty in the parameter estimates can be estimated and integrated over to determine the optimal policy under under uncertainty (Mangel 1985; Schapaugh and Tyre 2013). Uncertainty in the model structure itself can only be addressed in this approach by hypothesizing alternative model structures, and then performing some model choice or model averaging (B. K. Williams 2001; Athanassoglou and Xepapadeas 2012).

Parametric Models

We consider three candidate parametric models for the stock-recruitment function, which we refer to by the first authors of the publications in which they were first proposed.

We generate the data with a four-parameter model that contains a tipping point (an Allee effect, see [@Allen, @Courchamp]) below which the stock decreases to zero,

$$X_{t+1} = Z_t S_t e^{r\left(1-\frac{S_t}{K}\right)\left(\frac{S_t-\theta}{K}\right)}$$

$$S_t = X_t - h_t$$

The parameter C reflects the location of the tipping point, K the carrying capacity of the stock, and r the base recruitment rate. S_t represents the stock size after a harvest h_t has been implemented. Z_t represents a log-normal random variable of log-mean zero and log-standard deviation parameter σ.

We consider two alternative candidate models: the Ricker [@Ricker] stock-recruitment curve,

$$X_{t+1} = Z_t X_t e^{r\left(1-\frac{S_t}{K}\right)}$$

and an alternative four-parameter model adapted from @Myers,

$$X_{t+1} = Z_t \frac{r S_t^\theta}{1 - \frac{S_t^\theta}{K}}$$

which contains a tipping point for $\theta > 2$ and becomes a Beverton-Holt model at $\theta = 1$.

Bayesian Inference of Parametric models

Given the sample data, we infer posterior distributions for each of the three models listed above using a Markov Chain Monte Carlo Gibbs Sampler (jags, see appendix for implementation details and code) given uniform priors. We run six chains for 10^6 steps each and then assess convergence by Gelman-Rubin criterion and inspection of the traces, see appendix.

The Non-parametric Bayesian alternative for stock-recruitment curves

The use of Gaussian process (GP) regression (or "kriging" in the geospatial literature) to formulate a predictive model is relatively new in the context of modeling dynamical systems (Kocijan et al. 2005) and introduced in the ecological modeling and fisheries management by Munch et al. (2005). An accessible and thorough introduction to the formulation and use of GPs can be found in Rasmussen and Williams (2006).

The essence of the GP approach can be captured in the following thought experiment: An exhaustive parametric approach to the challenge of structural uncertainty might proceed by writing down all possible functional forms for the underlying dynamical system with all possible parameter values for each form, and then consider searching over this huge space to select the most likely model and parameters; or using a Bayesian approach, assign priors to each of these possible models and infer the posterior distribution of possible models. The GP approach can be thought of as a computationally efficient approximation to this approach. GPs represent a large class of models that can be though of as capturing or reasonably approximating the set of models in this collection. By modeling at the level of the process, rather than the level of parametric equation, we can more concisely capture the possible behavior of these curves. In place of a parametric model of the dynamical system, the GP approach postulates a prior distribution of (n-dimensional) curves that can be though of as approximations to a range of possible (parametric) models that might describe the data. The GP allows us to consider a set of possible curves simultaneously.

Background on Gaussian Process inference

Once the posterior Gaussian process (GP) has been estimated (e.g. see Munch et al. 2005), it is necessary to adapt it in place of the parametric equation for the stochastic dynamic programming (SDP) solution (see Mangel and Clark 1988 for a detailed description of parametric SDP methods) to the optimal policy. The essence of the idea is straight forward – we will use the estimated GP in place of the parametric growth function to determine

the stochastic transition matrix on which the SDP calculations are based. The SDP is solved in a discretized state space – both the continuously valued population densities X and harvest quotas h are first mapped to a bounded, discrete grid. (For simplicity we will consider a uniform grid, though for either parametric or GP-based SDP it is often advantageous to use a non-uniform discretization such as a basis function representation, e.g. see (Deisenroth, Rasmussen, and Peters 2009)).

The SDP approach then computes a transition matrix, $\mathbf{F}$. We demonstrate that calculation is just as straight forward based on the GP as it is in the classical context using the parametric model. The i, j of the transition matrix F entry gives the probability of transitioning into state x_i given that the system is in state x_j in the previous time-step. To generate the transition matrix based on the posterior GP, we need only the expected values at each grid point and the corresponding variances (the diagonal of the covariance matrix), as shown in Figure 1. Given the mean of the GP posterior at each grid-point as the vector E and variance at that point as vector V, the probability of transitioning from state x_i to state x_j is

$$\mathcal{N}\left(x_j|\mu = E_i, \sigma = \sqrt{V_i}\right)$$

where $\mathcal{N}$ is the Normal density at x_j with mean μ and variance σ^2. Strictly speaking, the transition probability should be calculated by integrating the normal density over the bin of width Δ centered at x_j. For a sufficiently fine grid that $f(x_j) \approx f(x_j + \Delta)$, it is sufficient to calculate the density at x_j and then row-normalize the transition matrix. The process can then be repeated for each possible discrete value of our control variable, (harvest h).

Pseudocode for the determining the transition matrix from the GP

```
for(h in h_grid)
  F_h = for(x_j in grid)
          for(i in 1:N)
            dnorm(x_j, mu[i]-h, V[i])
```

Using the discrete transition matrix we may write down the Bellman recursion defining the stochastic dynamic programming iteration:

$$V_t(x_t) = \max_h \mathbf{E}\left(h_t + \delta V_{t+1}(Z_{t+1} f(x_t - h_t))\right) \tag{1}$$

where $V(x_t)$ is the value of being at state x at time t, h is control (harvest level) chosen. Numerically, the maximization is accomplished as follows. Consider the set of possible control values to be the discrete values corresponding the grid of stock sizes. Then for each h_t there is a corresponding transition matrix $\mathbf{F}_h$ determined as described above but with mean $\mu = x_j - h_t$. Let $\vec{V_t}$ be the vector whose ith element corresponds to the value of having stock x_i at time t. Then let Π_h be the vector whose ith element indicates the profit from harvesting at

intensity h_t given a population x_i (e.g. $\max(x_i, h_t)$ since one cannot harvest more fish then the current population size). Then the Bellman recursion can be given in matrix form as

$$V_t = \max_h \left(\Pi_{h_t} + \delta \mathbf{F}_h V_{t+1}\right)$$

where the sum is element by element and the expectation is computed by the matrix multiplication $\mathbf{F}V_{t+1}$.

Pseudocode for the Bellman iteration

```
V1 <- sapply(1:length(h_grid), function(h){
      delta * F[[h]] %*% V +  profit(x_grid, h_grid[h])
    })
    # find havest, h that gives the maximum value
    out <- sapply(1:gridsize, function(j){
      value <- max(V1[j,], na.rm = T) # each col is a diff h, max over these
      index <- which.max(V1[j,])  # store index so we can recover h's
      c(value, index) # returns both profit value & index of optimal h.
    })
    # Sets V[t+1] = max_h V[t] at each possible state value, x
    V <- out[1,]                          # The new value-to-go
    D[,OptTime-time+1] <- out[2,]         # The index positions
```

This completes the algorithm adapting the GP to the sequential decision-making problem through SDP, which we believe has not yet been demonstrated in the literature. We provide an R package implementation of this, along with the Gaussian process inference, in the supplemental materials.

Estimating parametric models

We estimate posterior distributions for two parametric models: one using the structurally correct model as given in Eq (1), which we refer to as the "Parametric Bayes" model, and another using the familiar Ricker model, using a Gibbs sampler as described (with source code) in the appendix). In addition we estimate the parameters of the structurally correct model by maximum likelihood.

The posterior distribution for the hyper-parameters of the Gaussian process model are estimated by Metropolis-Hastings algorithm, again with details and code provided in the Appendix. Rasmussen and Williams (2006) provides an excellent general introduction to Gaussian Processes and Munch et al. (2005) first discusses their application in this context.

Results

```
require(MASS)
step_ahead <- function(x, f, p){
  h = 0
  x_predict <- sapply(x, f, h, p)
  n <- length(x_predict) - 1
  y <- c(x[1], x_predict[1:n])
  y
}
step_ahead_posteriors <- function(x){
gp_f_at_obs <- gp_predict(gp, x, burnin=1e4, thin=300)
df_post <- melt(lapply(sample(100),
  function(i){
    data.frame(time = 1:length(x), stock = x,
               GP = mvrnorm(1, gp_f_at_obs$Ef_posterior[,i], gp_f_at_obs$Cf_posterior[[i]]),
               True = step_ahead(x,f,p),
               MLE = step_ahead(x,f,est$p),
               Allen = step_ahead(x, allen_f, pardist[i,]),
               Ricker = step_ahead(x, ricker_f, ricker_pardist[i,]),
               Myers = step_ahead(x, myers_f, myers_pardist[i,]))
  }), id=c("time", "stock"))
}

df_post <- step_ahead_posteriors(x)

ggplot(df_post) + geom_point(aes(time, stock)) +
  geom_line(aes(time, value, col=variable, group=interaction(L1,variable)), alpha=.1) +
  scale_colour_manual(values=colorkey, guide = guide_legend(override.aes = list(alpha = 1)))
```

Figure 1 shows the mean inferred state space dynamics of each model relative to the true model used to generate the data, predicting the relationship between observed stock size (x-axis) to the stock size after recruitment the following year. All models except the MLE model estimate a distribution around the means shown here, and all models estimate a level of process noise, which is independent of the state value (x). Note that in contrast to the other models shown, the mean Gaussian process corresponds to a distribution of curves - as indicated by the gray band - which itself has a mean shown in black. Note that this mean GP is thus more certain of the dynamics in the region where data is available then where it is not.

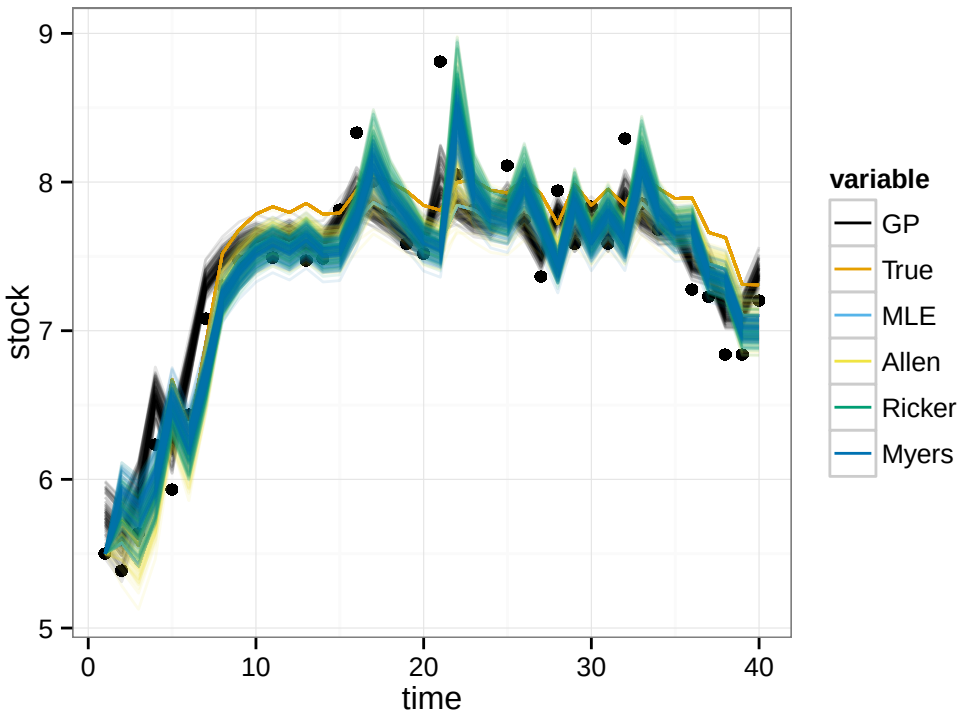

Figure 2: plot of chunk Figureb

While it would be natural (and straight forward) to condition the GP on passing through the origin (0,0) (see appendix), the estimate shown here is based only on the observed data. The observed data from which each model is estimated is also shown. The observations come from only a limited region of state space corresponding to unharvested or weakly harvested system. No observations occur at the theoretical optimum or near the tipping point.

```
policies <- melt(data.frame(stock=x_grid, sapply(OPT, function(x) x_grid[x])), id="stock")
names(policies) <- c("stock", "method", "value")

ggplot(policies, aes(stock, stock - value, color=method)) +
  geom_line(lwd=1.2, alpha=0.8) + xlab("stock size") + ylab("escapement")  +
  scale_colour_manual(values=colorkey)
```

The resulting optimal management strategy based on each of the inferred models is shown in Figure 2, against the optimal strategy given the true underlying dynamics. Policies are shown in terms of target escapement, S_t. Under models such as this a constant escapement policy is expected to be optimal (Reed 1979), whereby population levels below a certain size S are unharvested, while above that size the harvest strategy aims to return the population to S, resulting in the hockey-stick shaped policies shown.

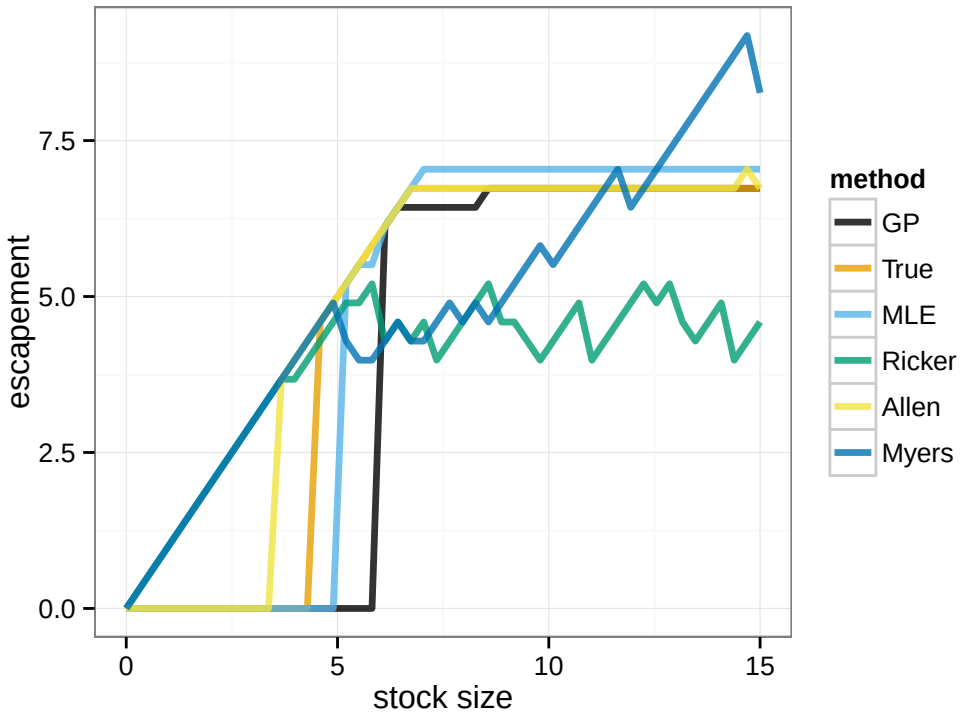

Figure 3: The steady-state optimal policy (infinite boundary) calculated under each model. Policies are shown in terms of target escapement, S_t, as under models such as this a constant escapement policy is expected to be optimal (Reed 1979).

```
ggplot(sims_data) +
  geom_line(aes(time, fishstock, group=interaction(reps,method), color=method), alpha=.1) +
  scale_colour_manual(values=colorkey, guide = guide_legend(override.aes = list(alpha = 1)))
```

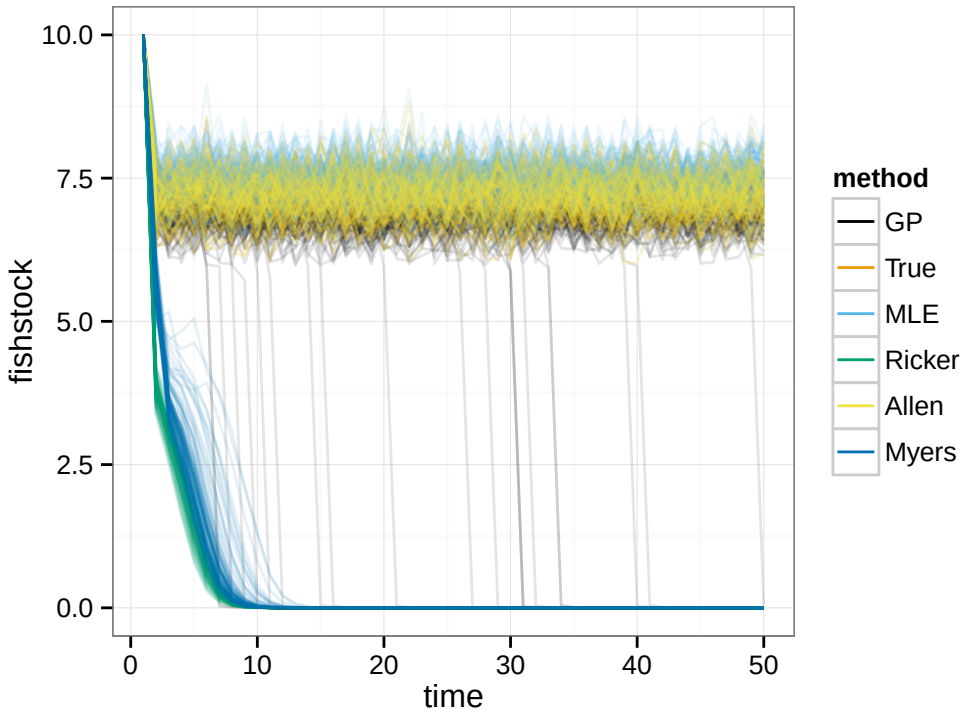

Figure 4: Gaussian process inference outperforms parametric estimates. Shown are 100 replicate simulations of the stock dynamics (eq 1) under the policies derived from each of the estimated models, as well as the policy based on the exact underlying model.

The consequences of managing 100 replicate realizations of the simulated fishery under each of the policies estimated is shown in Figure 3. As expected from the policy curves, the structurally correct model under-harvests, leaving the stock to vary around it's un-fished optimum. The structurally incorrect Ricker model over-harvests the population passed the tipping point consistently, resulting in the immediate crash of the stock and thus derives minimal profits.

The results shown in Figures 1-3 are not unique to the simulated data or models chosen here, but arises across a range of parameter values and simulations as shown in the supplemental figures. The results across this range can most easily be compared by the relative differences in net present value realized by each of the approaches, as shown in Figure 4. The Gaussian Process most consistently realizes a value close to the optimal solution, and importantly avoids ever driving the system across the tipping point, which results in the near-zero value cases in the parametric models.

```
ggplot(actual_over_optimal, aes(value)) + geom_histogram(aes(fill=variable)) +
```

```
  facet_wrap(~variable, scales = "free_y")  + guides(legend.position = "none") +
  xlab("Total profit by replicate") + scale_fill_manual(values=colorkey)
```

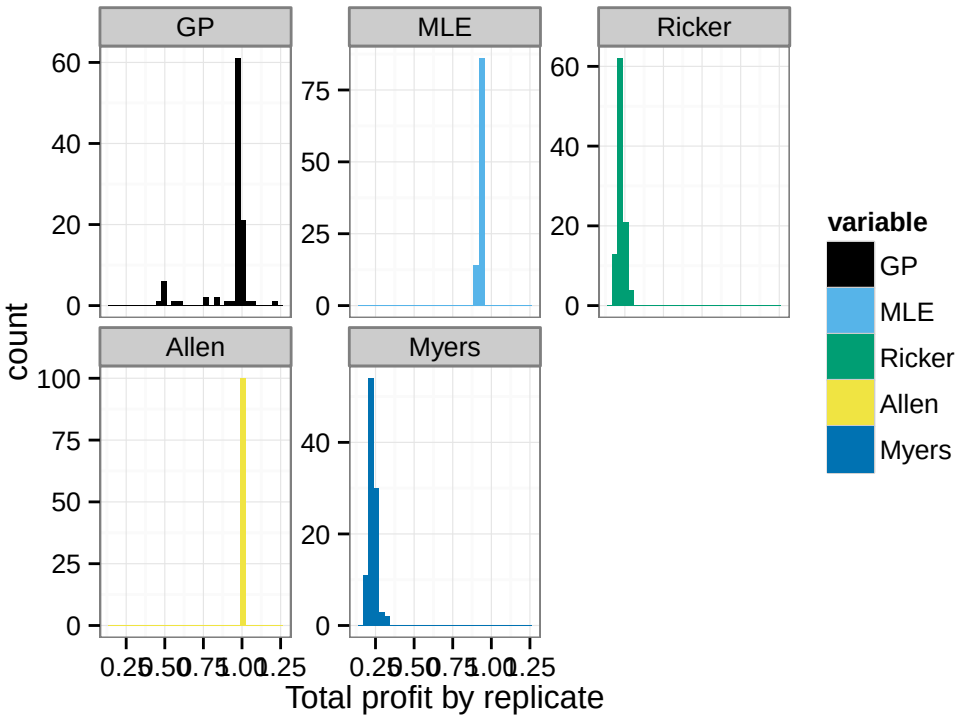

Figure 5: Histograms of the realized net present value of the fishery over a range of simulated data and resulting parameter estimates. For each data set, the three models are estimated as described above. Values plotted are the averages of a given policy over 100 replicate simulations. Details and code provided in the supplement.

```
ggplot(actual_over_optimal, aes(value)) + geom_histogram(aes(fill=variable), binwidth=0.1) +
  xlab("Total profit by replicate")+ scale_fill_manual(values=colorkey)
```

```
ggplot(actual_over_optimal, aes(value, fill=variable, color=variable)) +
  stat_density(aes(y=..density..), position="stack", adjust=3, alpha=.9) +
  xlab("Total profit by replicate")+ scale_fill_manual(values=colorkey)+ scale_color_manual(values=colorke
```

Discussion

Non-parametric Bayesian methods have received far too little attention in ecological modeling efforts that are aimed at improved conservation planning and decision making support. Such approaches may be particularly

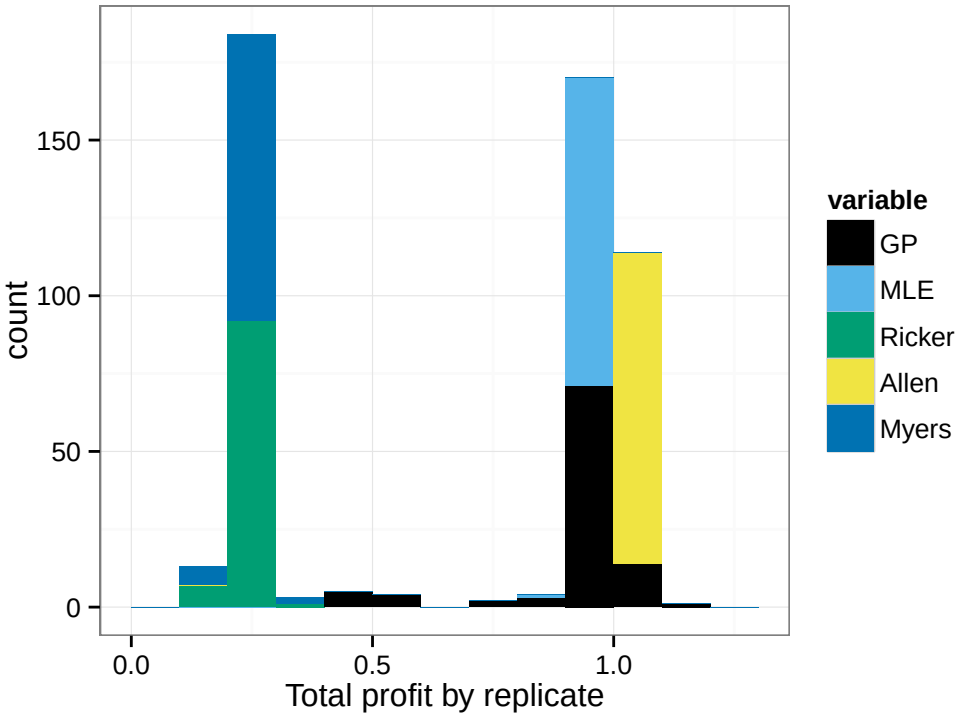

Figure 6: Histograms of the realized net present value of the fishery over a range of simulated data and resulting parameter estimates. For each data set, the three models are estimated as described above. Values plotted are the averages of a given policy over 100 replicate simulations. Details and code provided in the supplement.

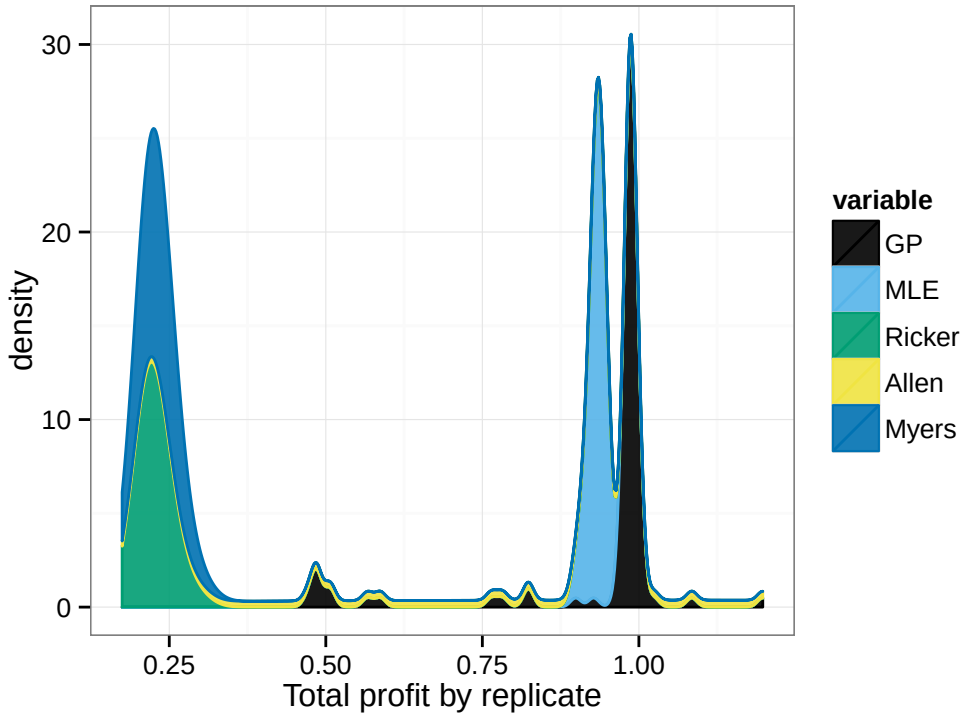

Figure 7: Histograms of the realized net present value of the fishery over a range of simulated data and resulting parameter estimates. For each data set, the three models are estimated as described above. Values plotted are the averages of a given policy over 100 replicate simulations. Details and code provided in the supplement.

useful when the available data is restricted to a limited area of state-space, which can lead parametric models to underestimate the uncertainty in dynamics at population levels (states) which have not been observed. One reason for the relative absence of nonparametric approaches in the natural resource management context may be the lack of existing approaches for adapting the non-parametric Bayesian models previously proposed (Munch et al. 2005) to a decision-theoretic framework. Adapting a non-parametric approach requires modification of existing methods for decision theory. We have illustrated how this might be done for a classic stochastic dynamic programming problem, opening the door for substantial further research into how these applications might be improved.

While non-parametric Bayesian approaches will not always be preferable to simple mechanistic models, we highlight three aspects of the problem consider here that make these methods particularly valuable. These aspects are common to many conservation decision making problems, which thus merit greater use of non-parametric approaches that can best take advantage of them.

1. Large uncertainty where the data is poor

The parametric models perform worst when they propose a management strategy outside the range of the observed data. The non-parametric Bayesian approach, in contrast, allows a predictive model that expresses a great deal of uncertainty about the probable dynamics outside the observed range, while retaining very good predictive accuracy in the range observed. The management policy dictated by the GP balance this uncertainty against the immediate value of the harvest, and act to stabilize the population dynamics in a region of state space in which the predictions can be reliably reflected by the data.

2. Predictive accuracy where data is good

While expressing larger uncertainty outside the observed data, the GP can also provide a better fit with smaller uncertainty inside the range of the observed data. This arises from the greater flexibility of the Gaussian process, which describes a large family of possible curves.
While in a parametric context this over-fitting would be more worrisome – a high-degree polynomial could fit the data even better – those concerns are driven by the resulting parametric fit outside the data, which may involve wild oscillations unsupported by the data. As we have seen in #1, the GP is less vulnerable to such unjustified predictions outside the data, and is meanwhile free to benefit from the greater fit where the data is available.

Future directions

In this simulated example, the underlying dynamics are truly governed by a simple parametric model, allowing the parametric approaches to be more accurate. Similarly, because the dynamics are one-dimensional dynamics and lead to stable nodes (rather than other attractors such as limit-cycles resulting in oscillations), the training

data provides relatively limited information about the dynamics. For these reasons, we anticipate that in higher-dimensional examples characteristic of ecosystem management problems that the machine learning approach will prove even more valuable.

Data complexity. Perhaps too far out of scope... The nonparametric Bayesian approach is also better suited to complex and disparate data. Incorporating various sources of information into mechanistic models can be an immensely difficult due to the increased complexity involved.

Online learning

In our treatment here we have ignored the possibility of learning during the management phase, in which the additional observations of the stock size could potentially improve parameter estimates. While we intend to address this possibility in future work in the context of these non-parametric models, we have not addressed it here for pedagogical reasons. In the context presented here, it is clear that the differences in performance arise from differences in the uncertainty inherent in the model formulations, rather than from differing abilities to learn. Because we consider a threshold system, online learning would not change this generic feature of a lack of data in a certain range of the state space which is better captured by the Gaussian process.

Acknowledgments

This work was partially supported by the Center for Stock Assessment Research, a partnership between the University of California Santa Cruz and the Fisheries Ecology Division, Southwest Fisheries Science Center, Santa Cruz, CA and by NSF grant EF-0924195 to MM.

Appendix

The appendices have not yet been assembled. Meanwhile, code to repeat the analyses, along with a complete log of all research conducted on this project, can be found at: https://github.com/cboettig/nonparametric-bayes

Markov Chain Monte Carlo Analysis

```
gp_assessment_plots[[1]]
```

```
gp_assessment_plots[[2]]
```

```
plot_allen_traces
```

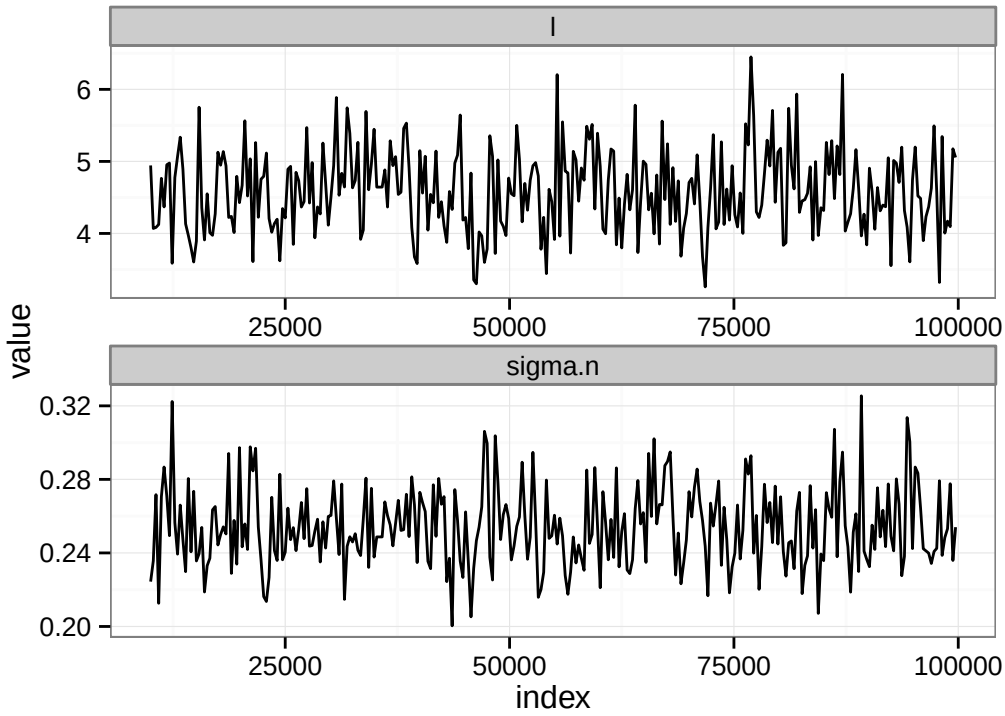

Figure 8: plot of chunk appendixplots

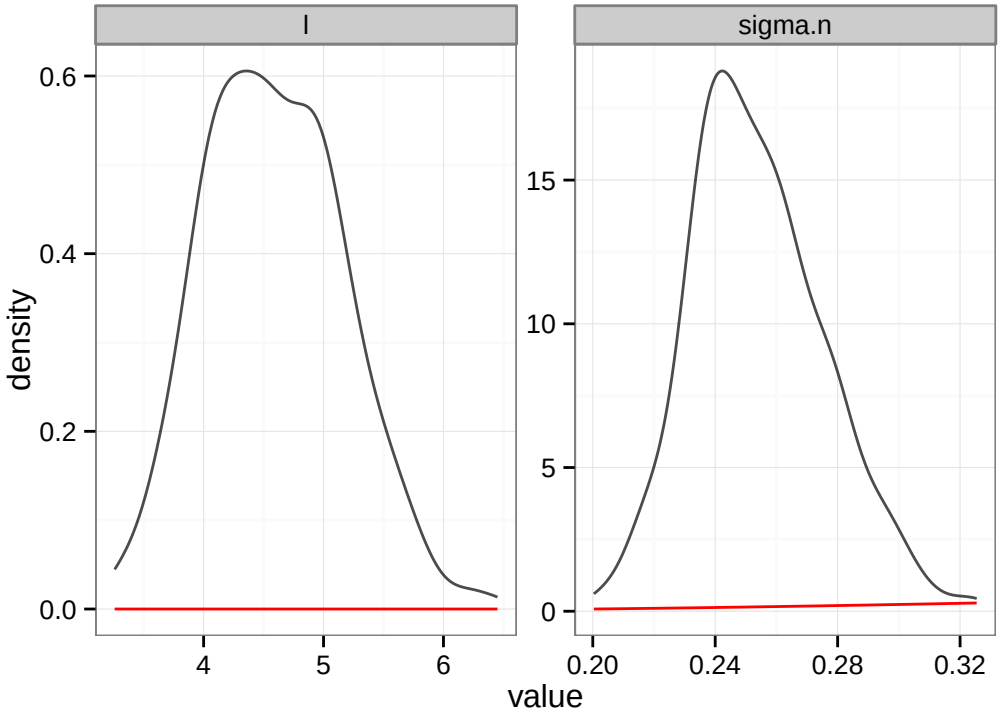

Figure 9: plot of chunk appendixplots

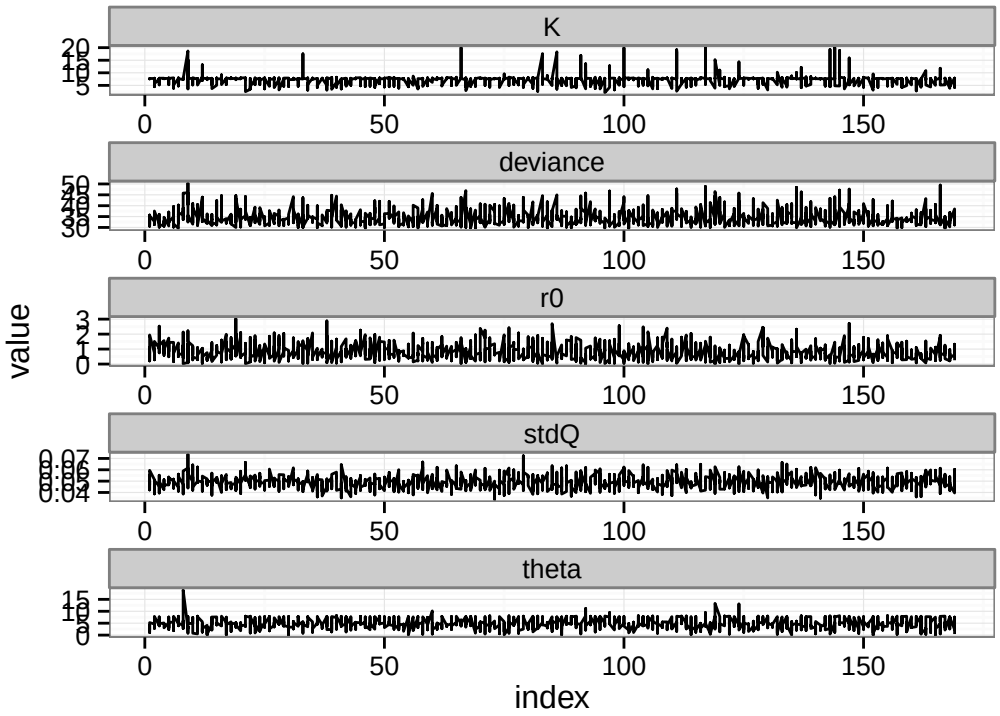

Figure 10: plot of chunk appendixplots

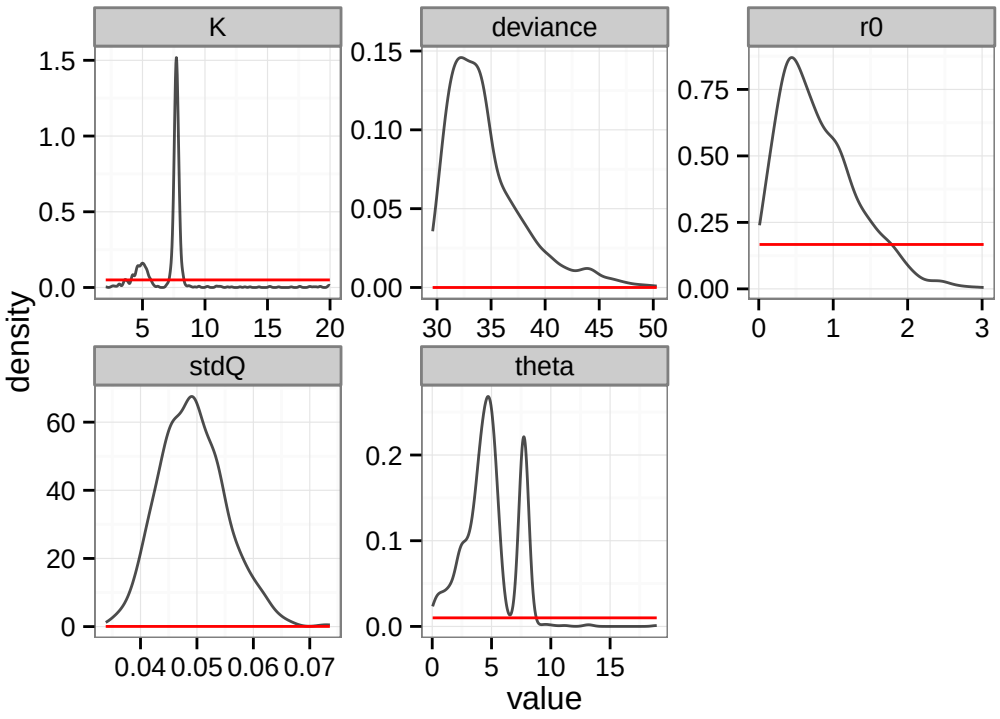

Figure 11: plot of chunk appendixplots

```
plot_allen_posteriors
```

```
plot_ricker_traces
```

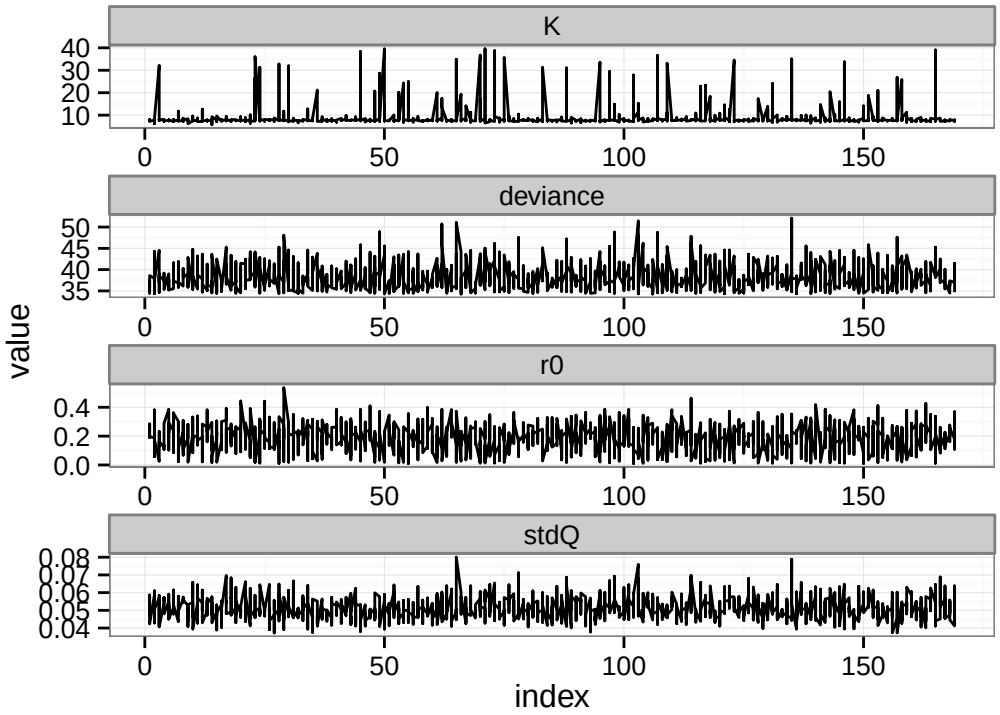

Figure 12: plot of chunk appendixplots

```
plot_ricker_posteriors
```

```
plot_myers_traces
```

```
plot_myers_posteriors
```

References

Allen, Linda J. S., Jesse F. Fagan, Göran Högnäs, and Henrik Fagerholm. 2005. "Population extinction in discrete-time stochastic population models with an Allee effect." *Journal of Difference Equations and Applications* 11 (4-5) (apr): 273–293. doi:10.1080/10236190412331335373. http://www.tandfonline.com/doi/abs/10.1080/10236190412331335373.

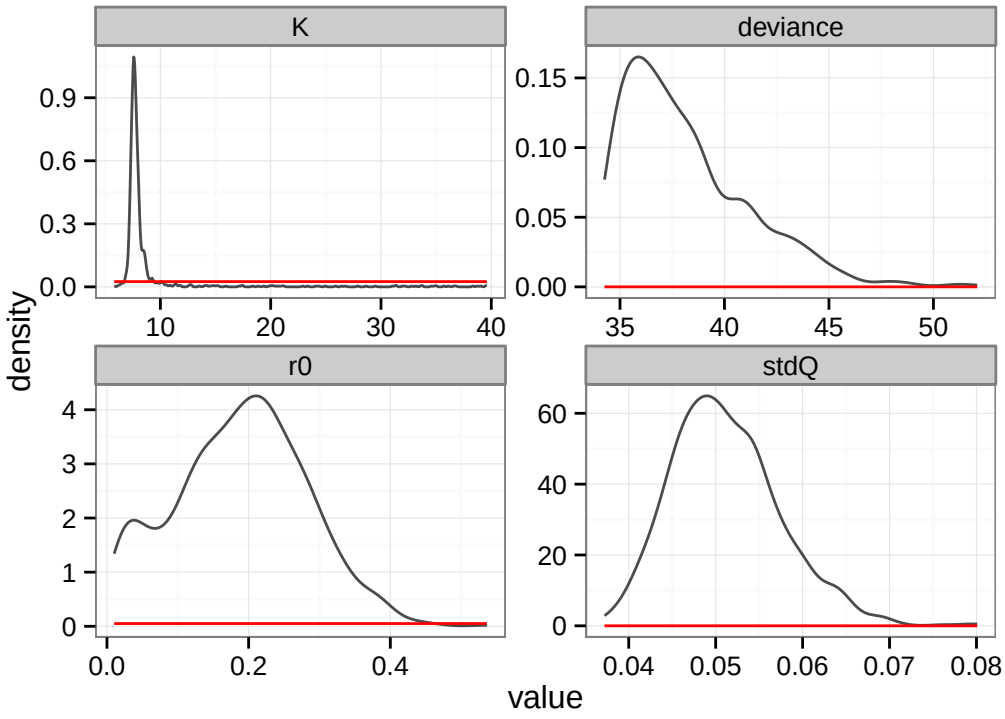

Figure 13: plot of chunk appendixplots

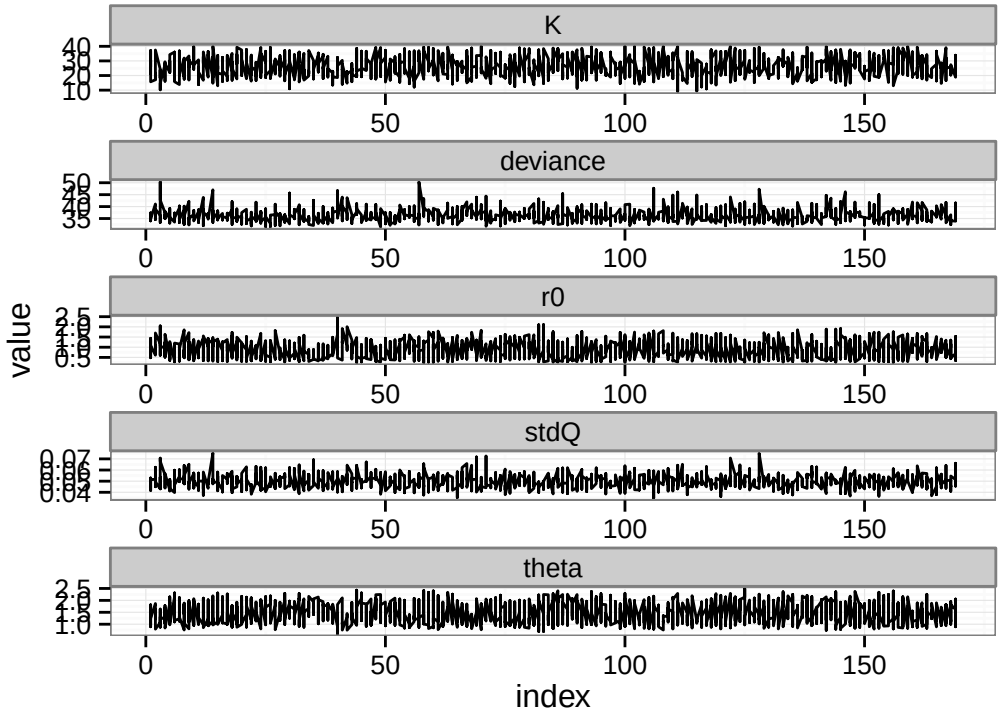

Figure 14: plot of chunk appendixplots

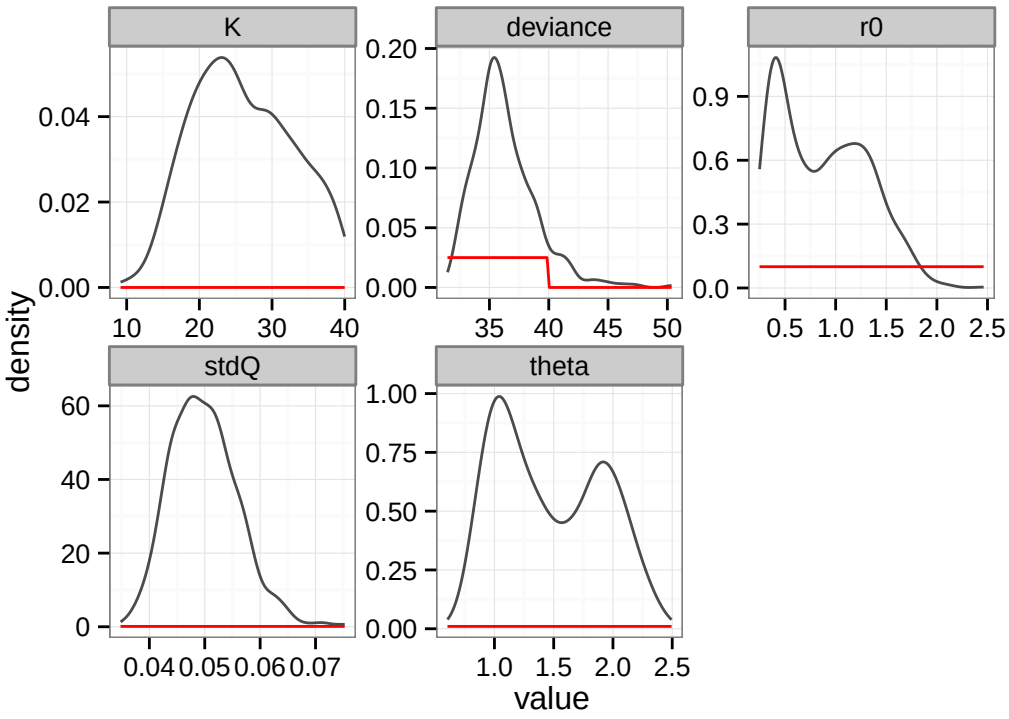

Figure 15: plot of chunk appendixplots

Athanassoglou, Stergios, and Anastasios Xepapadeas. 2012. "Pollution control with uncertain stock dynamics: When, and how, to be precautious." *Journal of Environmental Economics and Management* 63 (3) (may): 304–320. doi:10.1016/j.jeem.2011.11.001. http://linkinghub.elsevier.com/retrieve/pii/S0095069611001409.

Bestelmeyer, Brandon T., Michael C. Duniway, Darren K. James, Laura M. Burkett, and Kris M. Havstad. 2012. "A test of critical thresholds and their indicators in a desertification-prone ecosystem: more resilience than we thought." Ed. Katharine Suding. *Ecology Letters* (dec). doi:10.1111/ele.12045. http://doi.wiley.com/10.1111/ele.12045.

Brozović, Nicholas, and Wolfram Schlenker. 2011. "Optimal management of an ecosystem with an unknown threshold." *Ecological Economics* (jan): 1–14. doi:10.1016/j.ecolecon.2010.10.001. http://linkinghub.elsevier.com/retrieve/pii/S0921800910004167.

Courchamp, Franck, Ludek Berec, and Joanna Gascoigne. 2008. *Allee Effects in Ecology and Conservation*. Oxford University Press, USA. http://www.amazon.com/Effects-Ecology-Conservation-Franck-Courchamp/dp/0198570309.

Deisenroth, Marc Peter, Carl Edward Rasmussen, and Jan Peters. 2009. "Gaussian process dynamic programming." *Neurocomputing* 72 (7-9) (mar): 1508–1524. doi:10.1016/j.neucom.2008.12.019. http://linkinghub.elsevier.com/retrieve/pii/S0925231209000162.

Gordon, H. S. 1954. "The economic theory of a common-property resource: the fishery." *The Journal of Political Economy* 62 (2): 124–142. http://www.jstor.org/stable/10.2307/1825571.

Hilborn, Ray. 2007. "Reinterpreting the State of Fisheries and their Management." *Ecosystems* 10 (8) (oct): 1362–1369. doi:10.1007/s10021-007-9100-5. http://www.springerlink.com/index/10.1007/s10021-007-9100-5.

Hughes, Terry P., Cristina Linares, Vasilis Dakos, Ingrid a van de Leemput, and Egbert H. van Nes. 2013. "Living dangerously on borrowed time during slow, unrecognized regime shifts." *Trends in ecology & evolution* 28 (3) (mar): 149–55. doi:10.1016/j.tree.2012.08.022. http://www.ncbi.nlm.nih.gov/pubmed/22995893.

Kocijan, Juš, Agathe Girard, Bla\vz Banko, and Roderick Murray-Smith. 2005. "Dynamic systems identification with Gaussian processes." *Mathematical and Computer Modelling of Dynamical Systems* 11 (4) (dec): 411–424. doi:10.1080/13873950500068567. http://www.tandfonline.com/doi/abs/10.1080/13873950500068567.

Ludwig, Donald, and Carl J. Walters. 1982. "Optimal harvesting with imprecise parameter estimates." *Ecological Modelling* 14 (3-4) (jan): 273–292. doi:10.1016/0304-3800(82)90023-0. http://linkinghub.elsevier.com/retrieve/pii/0304380082900230.

Mangel, Marc. 1985. "Decision and control in uncertain resource systems:" 255. http://www.amazon.com/Decision-uncertain-resource-Mathematics-Engineering/dp/0124687202 http://dl.acm.org/citation.cfm?id=537497.

Mangel, Marc, and Colin W. Clark. 1988. *Dynamic Modeling in Behavioral Ecology*. Ed. John Krebs and Tim Clutton-Brock. Princeton: Princeton University Press.

May, Robert M., John R. Beddington, Colin W. Clark, Sidney J. Holt, and R. M. Laws. 1979. "Management of multispecies fisheries." *Science (New York, N.Y.)* 205 (4403) (jul): 267–77. doi:10.1126/science.205.4403.267. http://www.ncbi.nlm.nih.gov/pubmed/17747032.

Munch, Stephan B., Melissa L. Snover, George M. Watters, and Marc Mangel. 2005. "A unified treatment of top-down and bottom-up control of reproduction in populations." *Ecology Letters* 8 (7) (may): 691–695. doi:10.1111/j.1461-0248.2005.00766.x. http://doi.wiley.com/10.1111/j.1461-0248.2005.00766.x.

Rasmussen, Carl Edward, and C. K. I. Williams. 2006. *Gaussian Processes for Machine Learning*. Ed. Thomas Dietterich. Boston: MIT Press,. www.GaussianProcess.org/gpml.

Reed, William J. 1979. "Optimal escapement levels in stochastic and deterministic harvesting models." *Journal of Environmental Economics and Management* 6 (4) (dec): 350–363. doi:10.1016/0095-0696(79)90014-7. http://www.sciencedirect.com/science/article/pii/0095069679900147 http://linkinghub.elsevier.com/retrieve/pii/009506967990014

Schapaugh, Adam W., and Andrew J. Tyre. 2013. "Accounting for parametric uncertainty in Markov decision processes." *Ecological Modelling* 254 (apr): 15–21. doi:10.1016/j.ecolmodel.2013.01.003. http://linkinghub.elsevier.com/retrieve/pii/S0304380013000306.

Scheffer, Marten, Stephen R. Carpenter, J. A. Foley, C. Folke, and B. Walker. 2001. "Catastrophic shifts in

ecosystems." *Nature* 413 (6856) (oct): 591–6. doi:10.1038/35098000. http://www.ncbi.nlm.nih.gov/pubmed/11595939.

Williams, Byron K. 2001. "Uncertainty , learning , and the optimal management of wildlife." *Environmental and Ecological Statistics* 8: 269–288. doi:10.1023/A:1011395725123.

Worm, Boris, Edward B. Barbier, Nicola Beaumont, J. Emmett Duffy, Carl Folke, Benjamin S. Halpern, Jeremy B. C. Jackson, et al. 2006. "Impacts of biodiversity loss on ocean ecosystem services." *Science (New York, N.Y.)* 314 (5800) (nov): 787–90. doi:10.1126/science.1132294. http://www.ncbi.nlm.nih.gov/pubmed/17082450.

Worm, Boris, Ray Hilborn, Julia K. Baum, Trevor A. Branch, Jeremy S. Collie, Christopher Costello, Michael J. Fogarty, et al. 2009. "Rebuilding global fisheries." *Science (New York, N.Y.)* 325 (5940) (jul): 578–85. doi:10.1126/science.1173146. http://www.ncbi.nlm.nih.gov/pubmed/19644114.