Appendix 1 – JAGS Model code for replicating Models 1-4 with priors from Andrews et al. 2012

# Model 1

model{

for (i in 1:N) {

# for (j in 2:n[i]) {

for (j in n[i]) {

L[i, j] ~ dnorm(L\_Exp[i, j], tau)

L\_Exp[i, j] <- Linf[i] \*(1.0 - exp(-k[i]\*(A[i]+dt[i, j -1])))

# posterior prediction

L.pred[i, j] ~ dnorm(L\_Exp[i, j], tau)

p.value[i, j] <- step(L.pred[i, j] - L[i, j])

}

L[i, 1] ~ dnorm(L\_Exp[i, 1], tau)

L\_Exp[i, 1] <- Linf[i] \*(1.0 - exp(-k[i]\*A[i]))

# posterior prediction

L.pred[i, 1] ~ dnorm(L\_Exp[i, 1], tau)

p.value[i, 1] <- step(L.pred[i, 1]- L[i, 1])

Linf[i] ~ dnorm(Linf\_mu, Linf\_tau)

k[i] ~ dnorm(k\_mu, k\_tau) T(0,1)

A[i] ~ dgamma(Shape, rate)

}

Linf\_std <- sqrt(1/Linf\_tau)

k\_std <- sqrt(1/k\_tau)

variance <- 1/tau

Linf\_mu ~ dnorm(67.5, 0.0001)

Linf\_tau ~ dgamma(0.308642, 0.0001)

Shape ~ dunif(0, 100)

rate ~ dunif(0, 100)

k\_mu ~ dbeta(0.242, 1) T(0.01,0.9)

k\_tau ~ dgamma(307.787, 0.0001)

tau ~ dgamma(0.001, 0.0001)

}'

# Model 2

model{

for (i in 1:N) {

# for (j in 2:n[i]) {

for (j in n[i]) {

L[i, j] ~ dnorm(L\_Exp[i, j], tau)

L\_Exp[i, j] <- Linf[i] \*(1.0 - exp(-k\*(A[i]+dt[i, j -1])))

L.pred[i, j] ~ dnorm(L\_Exp[i, j], tau)

p.value[i, j] <- step(L.pred[i, j] - L[i, j])

}

L[i, 1] ~ dnorm(L\_Exp[i, 1], tau)

L\_Exp[i, 1] <- Linf[i] \*(1.0 - exp(-k\*A[i]))

L.pred[i, 1] ~ dnorm(L\_Exp[i, 1], tau)

p.value[i, 1] <- step(L.pred[i, 1]- L[i, 1])

Linf[i] ~ dnorm(Linf\_mu, Linf\_tau)

A[i] ~ dgamma(Shape, rate)

}

Linf\_std <- sqrt(1/Linf\_tau)

k\_std <- sqrt(1/k\_tau)

variance <- 1/tau

k ~ dnorm(k\_mu, k\_tau)

Linf\_mu ~ dnorm(67.5, 0.0001)

Linf\_tau ~ dgamma(0.308642, 0.0001)

Shape ~ dunif(0, 100)

rate ~ dunif(0, 100)

k\_mu ~ dbeta(0.242, 1) T(0.01,0.9)

k\_tau ~ dgamma(307.787, 0.0001)

tau ~ dgamma(0.001, 0.0001)

}

# Model 3

model{

for (i in 1:N) {

# for (j in 2:n[i]) {

for (j in n[i]) {

L[i, j] ~ dnorm(L\_Exp[i, j], tau)

L\_Exp[i, j] <- Linf\*(1.0 - exp(-k[i]\*(A[i]+dt[i, j -1])))

L.pred[i, j] ~ dnorm(L\_Exp[i, j], tau)

p.value[i, j] <- step(L.pred[i, j] - L[i, j])

}

L[i, 1] ~ dnorm(L\_Exp[i, 1], tau)

L\_Exp[i, 1] <- Linf \*(1.0 - exp(-k[i]\*A[i]))

L.pred[i, 1] ~ dnorm(L\_Exp[i, 1], tau)

p.value[i, 1] <- step(L.pred[i, 1]- L[i, 1])

k[i] ~ dnorm(k\_mu, k\_tau) T(0,1)

A[i] ~ dgamma(Shape, rate)

}

Linf\_std <- sqrt(1/Linf\_tau)

k\_std <- sqrt(1/k\_tau)

variance <- 1/tau

Linf ~ dnorm(Linf\_mu, Linf\_tau)

Linf\_mu ~ dnorm(67.5, 0.0001)

Linf\_tau ~ dgamma(0.308642, 0.0001)

Shape ~ dunif(0, 100)

rate ~ dunif(0, 1000)

k\_mu ~ dbeta(0.242, 1) T(0.01,0.9)

k\_tau ~ dgamma(307.787, 0.0001)

tau ~ dgamma(0.01, 0.0001)

}

# Model 4

model{

for (i in 1:N) {

# for (j in 2:n[i]) {

for (j in n[i]) {

L[i, j] ~ dnorm(L\_Exp[i, j], tau)

L\_Exp[i, j] <- Linf\*(1.0 - exp(-k\*(A[i]+dt[i, j-1])))

L.pred[i, j] ~ dnorm(L\_Exp[i, j], tau)

p.value[i, j] <- step(L.pred[i, j] - L[i, j])

}

# Predicting length at capture

L[i, 1] ~ dnorm(L\_Exp[i, 1], tau)

L\_Exp[i, 1] <- Linf \*(1.0 - exp(-k\*A[i]))

L.pred[i, 1] ~ dnorm(L\_Exp[i, 1], tau)

p.value[i, 1] <- step(L.pred[i, 1]- L[i, 1])

A[i] ~ dgamma(Shape, rate)

}

k\_std <- sqrt(1/k\_tau)

variance <- 1/tau

k ~ dnorm(k\_mu, k\_tau)

Linf ~ dnorm(Linf\_mu, Linf\_tau)

Linf\_mu ~ dnorm(67.5, 0.0001)

Linf\_tau ~ dgamma(0.308642, 0.0001)

Linf\_std <- sqrt(1/Linf\_tau)

Shape ~ dunif(0, 100)

rate ~ dunif(0, 100)

k\_mu ~ dbeta(0.242, 1) T(0.01,0.9)

k\_tau ~ dgamma(307.787, 0.0001)

tau ~ dgamma(0.001, 0.0001)

}