Model checks

Chris Hoover

March 25, 2019

Initial model simulations

R_0 Check

First estimate R_0 from initial parameters then get equilibirum values of state variables for same parameter set

```
r0.Ag(parameters = init_pars)[3]
```

```
##       R0
## 6.170588
```

```
r0.Base(init_pars)
```

```
## [1] 6.629723
```

```
r0.pred(init_pars)[3]
```

```
##  R0_pred
## 6.170588
```

```
base.start = c(S = 40*area,
               E = 0,
               I = 0,
               W = 2)

base_pred.start = c(S = 40*area,
                    E = 0,
                    I = 0,
                    W = 2,
                    P = 0)

base_eq <- runsteady(y = base.start, func = base_mod, parms = init_pars)$y

base_eq
```

```
##         S         E         I         W
##  200.2263 4590.0820 1147.5205   44.6918
```

```
eq_vals <- runsteady(y = base_pred.start, func = agrochem_mod, parms = init_pars)$y

eq_vals
```

```
##          S          E          I          W          P
##  200.22631 4590.08204 1147.52051   44.69179    0.00000
```

```
pred_vals <- runsteady(y = base_pred.start, func = pred_mod, parms = init_pars)$y

pred_vals
```

```
##          S          E          I          W          P
##  200.22631 4590.08204 1147.52051   44.69179    0.00000
```

```
pred.start = c(S = 40*area,
             E = 0,
             I = 0,
             W = 2,
             P = 0.1*area)

eq_vals <- runsteady(y = pred.start, func = agrochem_mod, parms = init_pars)$y

eq_vals
```

```
##          S          E          I          W          P
##  286.29019 3748.16267  841.54722   32.77524   16.88034
```

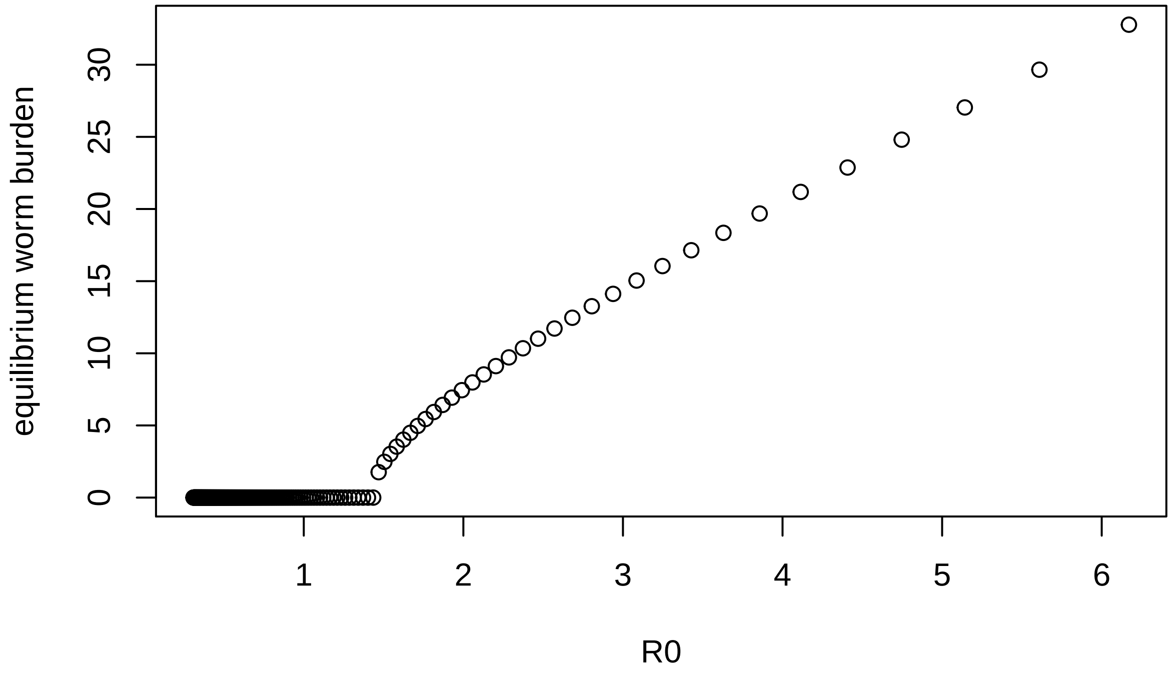
```
pred_vals <- runsteady(y = pred.start, func = pred_mod, parms = init_pars)$y

pred_vals
```

```
##          S          E          I          W          P
##  286.29019 3748.16267  841.54722   32.77524   16.88034
```

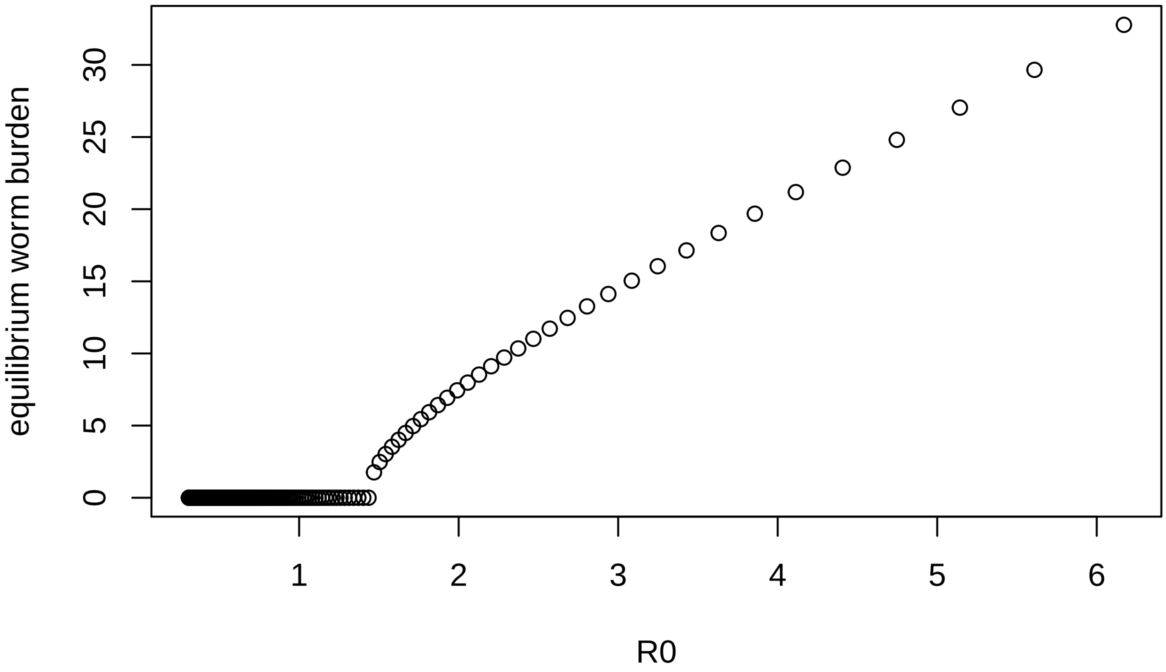
R_0 and dynamics check

Ensure that equilibrium worm burden is in qualitative agreement with estimates of R_0 (i.e. not getting rampant infection when $R_0 < 1$)

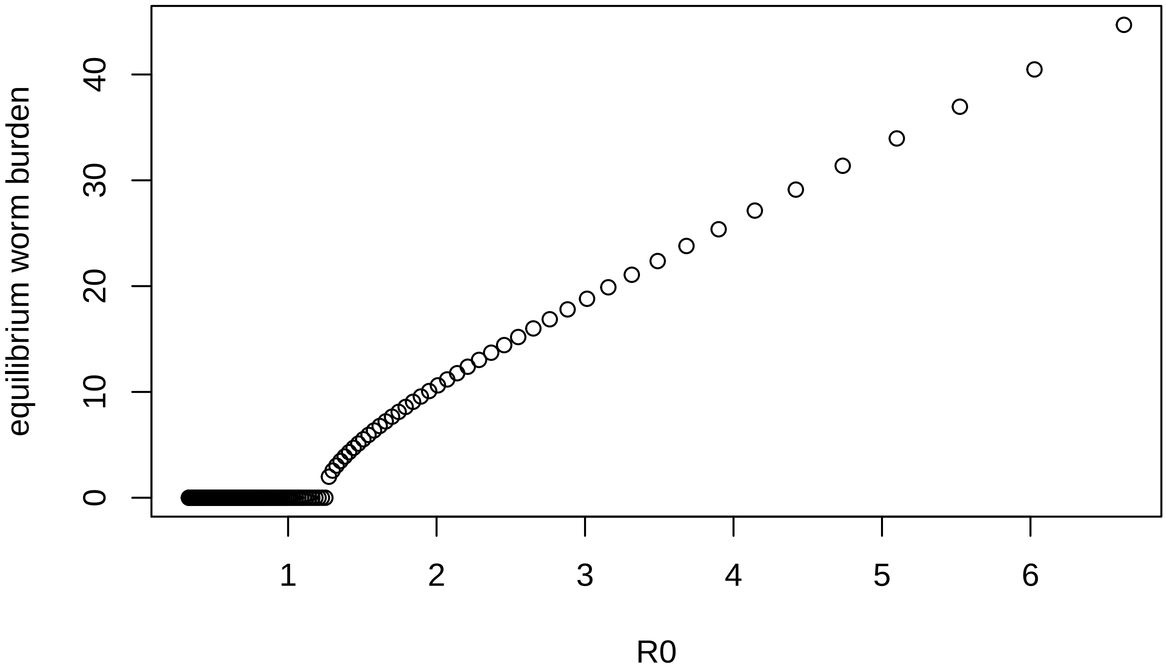
```
test_pars <- init_pars

test_r0s <- numeric()
test_ws <- numeric()

pred_r0s <- numeric()
pred_ws <- numeric()

base_r0s <- numeric()
base_ws <- numeric()

for(i in seq(1, 20, 0.1)){
  test_pars["beta"] <- init_pars["beta"]/i
  test_pars["lambda"] <- init_pars["lambda"]/i

  test_r0s[i*10-9] <- r0.Ag(parameters = test_pars)[3]
  test_ws[i*10-9] <- runsteady(y = pred.start, func = agrochem_mod, parms = test_pars)$y["W"]

  pred_r0s[i*10-9] <- r0.pred(parameters = test_pars)[3]
  pred_ws[i*10-9] <- runsteady(y = pred.start, func = pred_mod, parms = test_pars)$y["W"]

  base_r0s[i*10-9] <- r0.Base(parameters = test_pars)
  base_ws[i*10-9] <- runsteady(y = base.start, func = base_mod, parms = test_pars)$y["W"]

  #print(i*10-9)
}

plot(test_r0s, test_ws,
     xlab = "R0", ylab = "equilibrium worm burden")
```



```
plot(pred_r0s, pred_ws,
     xlab = "R0", ylab = "equilibrium worm burden")
```



```r
plot(base_r0s, base_ws,
     xlab = "R0", ylab = "equilibrium worm burden")
```



Looks good. Since R_0 doesn't consider the mating function, it make sense that R_0 has to be slightly above 1 for infection to occur. Furthermore, with the addition of predators, the threshold becomes even higher due to predation on infected and exposed snails that further reduces transmission