How much visits do we need?

Abstract

While the positive effect of pollinators on crop production has been clearly demonstrated, crop level predictions of how pollination rates increase crops reproductive success remains elusive and requires time consuming experiments. Defining the functional form of this relationships is of pivotal importance to understand when pollination is limiting, and develop operational visitation rate values that can be used by practitioners to rapidly assess pollination levels. Given the large variety of crops, and varieties, running this experiments for all of them is unlikely to be achieved. Here we propose and test a generalized model based on biological first principles to infer optimal pollinator visitation rates for multiple crops. First we show that a non-inflicted functional form fits well empirical data. Second, we develop a flexible model to produce this functional forms from easy to obtain plant and pollinator traits such as selfing rates, ovule number and single visit effective pollen deposition and test its sensitivity to the certainty in those initial values. Finally, we apply our framework to X case studies to show the advantages and limitations of the framework.

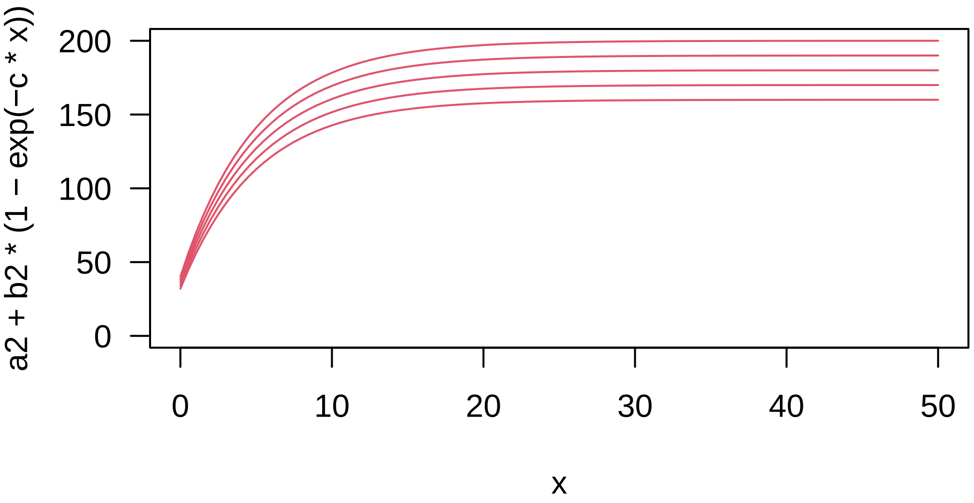
Mechanistic model

We test the non-inflicted curve proposed by Morris et al. 2004. The advantage of this functional form is that its parameters have biological meaning. The curve is defined by three parameters which govern the relationship between pollinator visitation rates and crop seed set. Parameter (a) sets the y-intercept (i.e., seed set for unvisited flowers), parameter (b) sets the asymptote (i.e., maximum seed set attainable by a single flower), and parameter (c) governs the rate of approach to the asymptote.

First we load the functions.

Second we can visualize the shape of the curve:

```r
plot_visits(a = 20, b = 200, c = 0.2, from_ = 0, to_ = 50, add = FALSE)
plot_visits(a = 20, b = 190, c = 0.2, from_ = 0, to_ = 50, add = T)
plot_visits(a = 20, b = 180, c = 0.2, from_ = 0, to_ = 50, add = T)
plot_visits(a = 20, b = 170, c = 0.2, from_ = 0, to_ = 50, add = T)
plot_visits(a = 20, b = 160, c = 0.2, from_ = 0, to_ = 50, add = T)
```



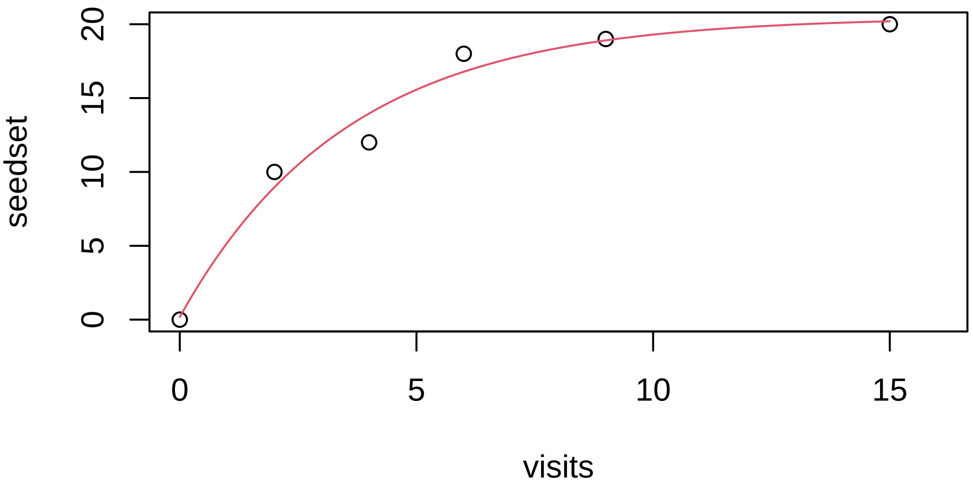
```r
#check what happens on the intercept a (a is in %??)
```

Third, we can fit the a, b and c params if we have a dataset relating visitation rates to seed set. Mike might have those.

```r
#Create a fake dataset
visits <- c(0,2,4,6,9,15)
seedset <- c(0,10,12,18,19,20)
(mf <- fit_data(seedset = seedset, visitation = visits))
```

```
        a          b          c 
0.8754890 20.4875160  0.2836655 
```

```
#ADD SE to output and plots.
#Also add option to estimate only c, or only b and c.
#Test fit
plot(seedset ~ visits, xlim = c(0,16), ylim = c(0, 20))
plot_visits(mf["a"], mf["b"], mf["c"], to_ = 15, add_ = TRUE)
```



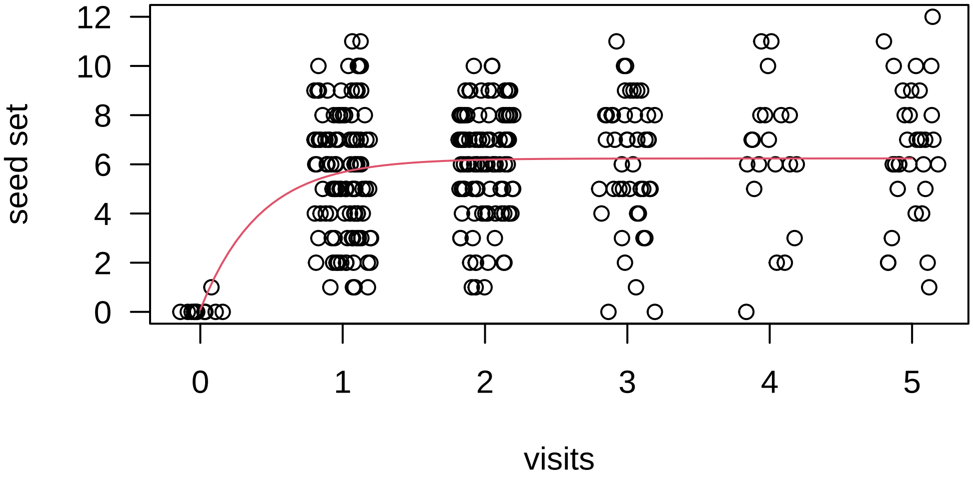
```
#We can compare to cuadratic, linear and ...
```

We obtain a nice fit of the fake dataset.

```
#try with mikes apple data
apple <- read.csv(file = "data/apple.csv")
#head(apple)
#unique(apple$Pollinator) #[1] "Bombus terrestris"    "Apis mellifera"       #"Episyrphus bal
apple <- subset(apple, Pollinator %in% c("Apis mellifera", "Osmia bicornis", "Control"))
(mf <- fit_data(seedset = apple$SeedNumber, visitation = apple$VisitNumber))
```

```
       a        b        c
1.481213 6.239269 2.393663
```

```
#Test fit
plot(apple$SeedNumber ~ jitter(apple$VisitNumber), las = 1, xlab = "visits", ylab = "seed se
plot_visits(mf["a"], mf["b"], mf["c"], to_ = 5, add_ = TRUE)
```

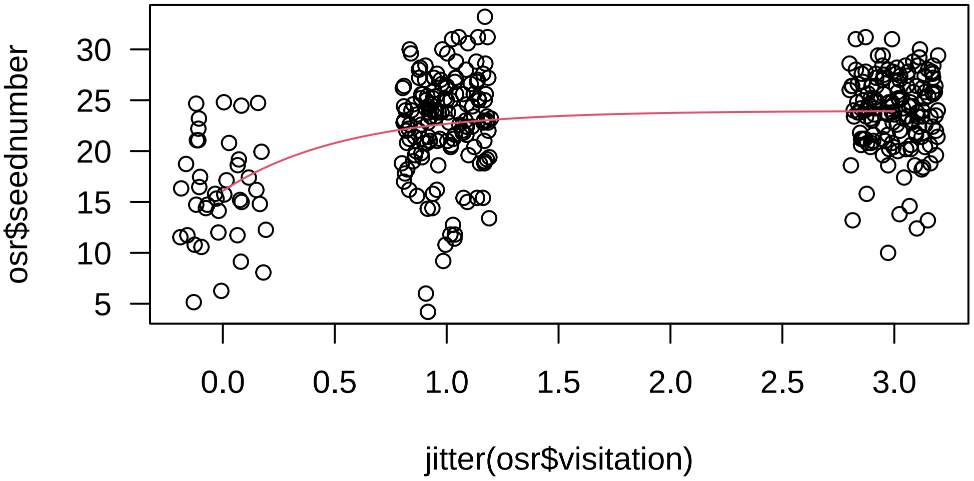


Apple rapidly saturates, but variation is huge

```
#try with mikes osr data
osr <- read.csv(file = "data/osr.csv")
#head(osr)
osr <- subset(osr, pollinator != "Hand")
#unique(osr$visitation)
#osr[which(is.na(osr$visitation)),]
osr$visitation <- as.factor(osr$visitation)
levels(osr$visitation) <- c("3", "1")
osr$visitation <- as.character(osr$visitation)
osr$visitation[which(is.na(osr$visitation))] <- 0
osr$visitation <- as.numeric(osr$visitation)
(mf <- fit_data(seedset = osr$seednumber, visitation = osr$visitation))
```

```
        a         b         c 
67.158393 23.957914  1.816795 
```

```
#Test fit
plot(osr$seednumber ~ jitter(osr$visitation), las = 1)
plot_visits(mf["a"], mf["b"], mf["c"], to_ = 3, add_ = TRUE)
```

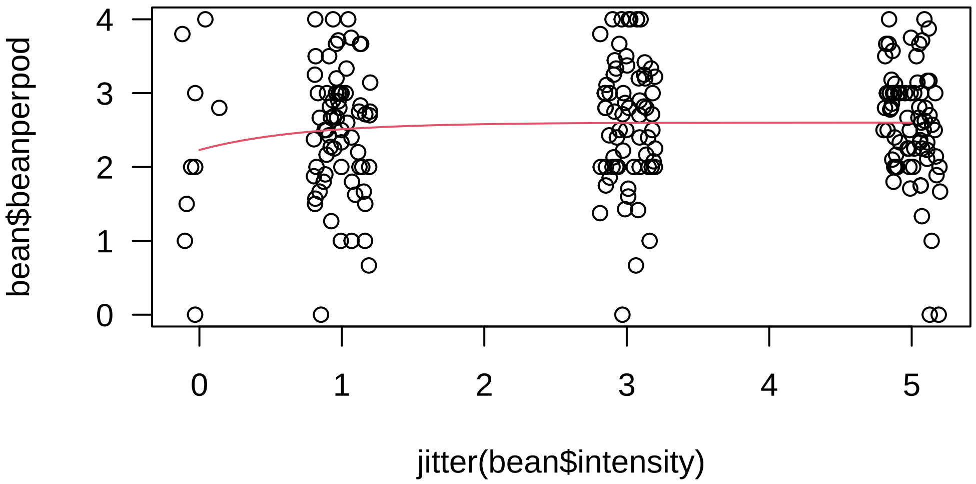


OSR also saturates very quickly…

```
#try with mikes bean data
bean <- read.csv(file = "data/bean.csv")
#head(bean)
#unique(bean$pollinator) #What is OR?
bean <- subset(bean, pollinator != "Hand")
#unique(bean$intensity)
#bean[which(is.na(bean$intensity)),]
bean$intensity <- as.factor(bean$intensity)
levels(bean$intensity) <- c("5", "1", "3")
bean$intensity <- as.character(bean$intensity)
bean$intensity[which(is.na(bean$intensity))] <- 0
bean$intensity <- as.numeric(bean$intensity)
(mf <- fit_data(seedset = bean$beanperpod, visitation = bean$intensity))
```

```
        a         b         c 
85.722084  2.601975  1.406839 
```

```
#Test fit
plot(bean$beanperpod ~ jitter(bean$intensity), las = 1)
plot_visits(mf["a"], mf["b"], mf["c"], to_ = 5, add_ = TRUE)
```

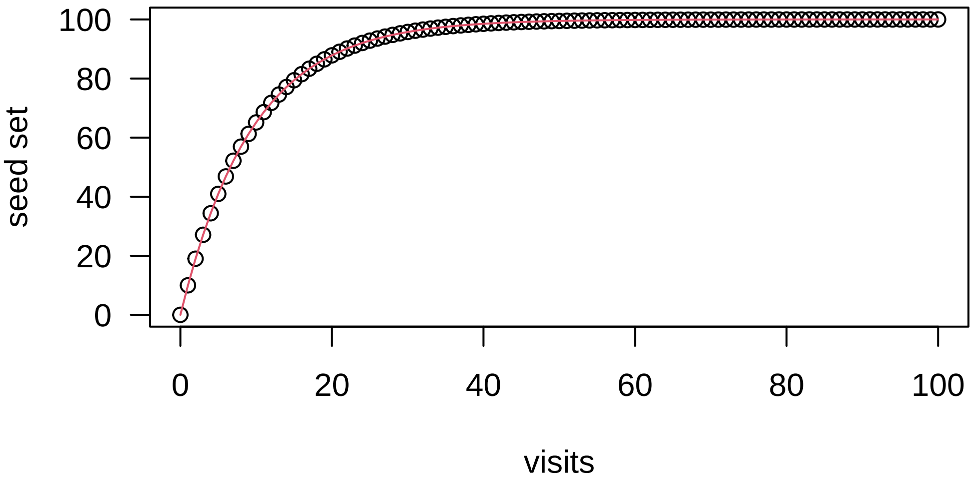


Beans they don't care about pollinators.

Fourth, we derive a generalized version that rely on first principles and does not require hard won data. To that end, we set b as the maximum seed set a flower can attain (see below for plant or plot level metrics). This can be extracted from the crop basic biology, as it will match the ovule production, a parameter often easy to obtain from the literature or to simply obtain an average by counting ovules/seeds in the field. Parameter a can be approximated knowing plants selfing rates. That is, who many seeds the flower produce under no pollination. This information is available in the literature (e.g. Klein et al. 2007) or can be calculated with basic experiments. The trickier parameter to calculate is c. For this, we rely on pollinator per visit efficiency. Many studies measure seed set, pollen tube development or pollen deposition after a single visit. This values (especially the first two values) can be used to extrapolate how many visits a flower requires until fertilizing all its ovules. The average pollinator efficiency can be calculated across pollinators, which is the representative metric. This data is already collected for many crops, (is expected to vary less among crops and pollinators See Vazquez et al. 2005) and can be collected in the field if necessary. However, it is known that each subsequent visit is less efficient (Kendall et al.), hence we can penalize each consecutive visit by a %. Which % can be informed by data in the few crops for which this information exists, and sensitivity analysis can be done in the importance of this parameter (see below).

With this, we can explore different scenarios:

```
calculate_visits(a = 0, b = 100, SVD = 10, loss = 0.1, from_ = 0, to_ = 100,
                                               plot_ = TRUE, col_ = 2)
```

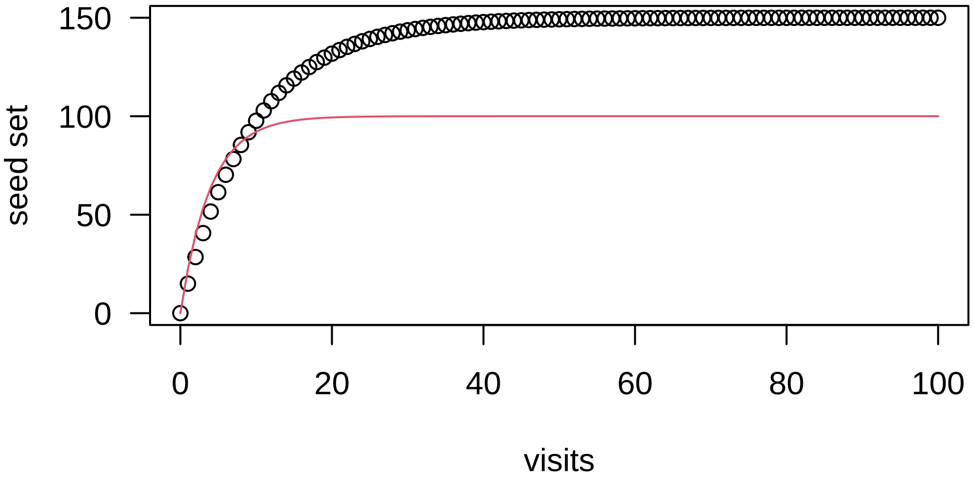


```
          c
0.1053605
```

```
#we can calculate loss as % decrease from linea relationship
```

A crop that do not produce any seed without pollinators, has 100 ovules and pollinators fertilize a mean of 10 pollen grains per visit, with a penalization of 10%. In this case the black dots represents the pollen grain deposition as the number of visits increases. The red line is the fit of the mechanistic model. In this case, they both reflect the same story. The interpretation is that 30 visits per flower is optimal.

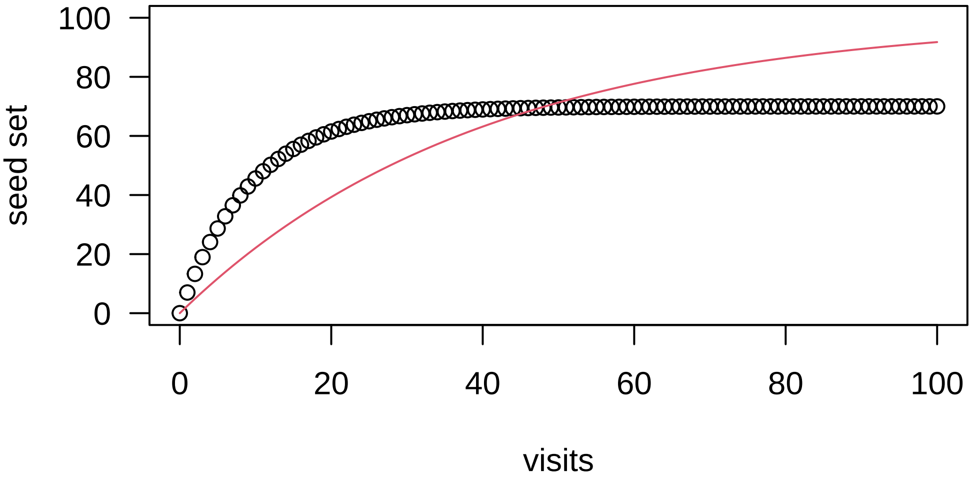
```
calculate_visits(a = 0, b = 100, SVD = 15, loss = 0.1, from_ = 0, to_ = 100,
                                               plot_ = TRUE, col_ = 2)
```



```
        c
0.254383
```

Here, all params are the same, but average pollen deposition is higher. Hence, seed set (red line) saturates faster than pollination, which overshoots. Optimal visits is ~15 and beyond this number, we risk adding too many pollen, which can have different consequences depending on the crop (e.g. damage or pollen interference).

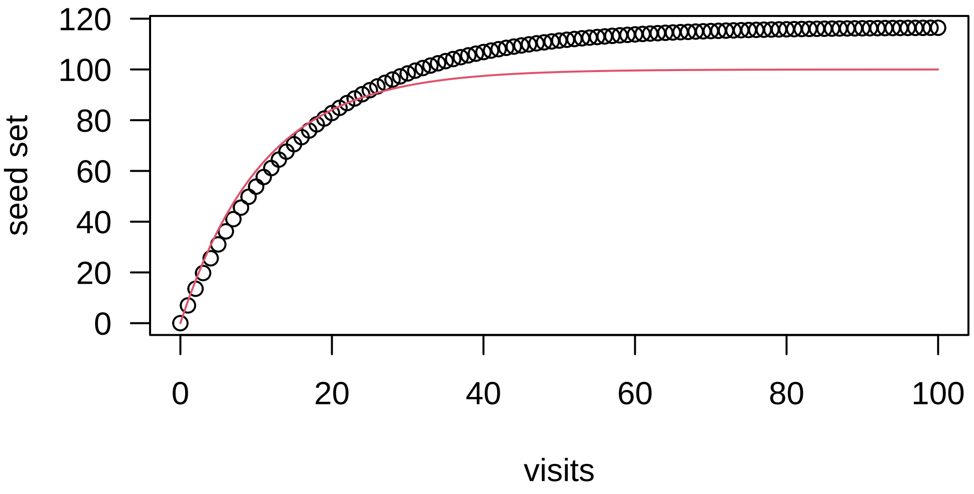
```
calculate_visits(a = 0, b = 100, SVD = 7, loss = 0.1, from_ = 0, to_ = 100,
                                 plot_ = TRUE, col_ = 2)
```



```
         c
0.02496434
```

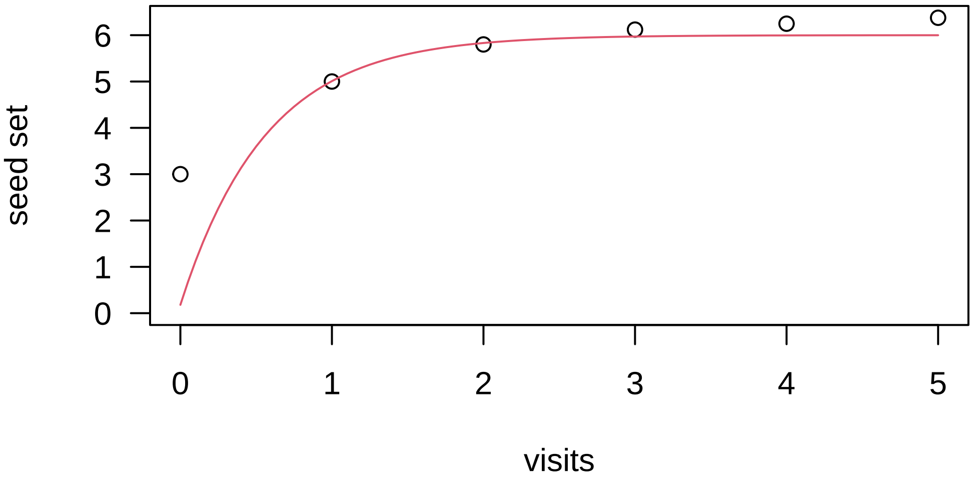
Here, in contrast, average pollen deposition is lower (7). I think this situation is unlikely to happen in nature, because it indicates natural pollination will never be optimal. I think this indicates the model is mis-specified. For example, we migh be underestimating single visit efficiencies, or penalizing too much consecutive visits. Let's see this last option:

```
calculate_visits(a = 0, b = 100, SVD = 7, loss = 0.06, from_ = 0, to_ = 100,
                             plot_ = TRUE, col_ = 2)
```



```
          c
0.09185109
```

```
#try with mikes data
calculate_visits(a = 3, b = 6, SVD = 2, loss = 0.6, from_ = 0, to_ = 5,
                              plot_ = TRUE, col_ = 2)
```



```
       c
1.770212
```

```
#why a and the plotted intercept are not equal?
```

Yes, if we reduce the penalty to a 6%, we have plausible numbers again.

Caveats:

- We assume optimal conditions of water, light, nutrients, and no other limiting factors. This is why linking seed set to yield is tricky. However, if yield is reduced, while pollination is optimal, we know other factors are limiting for ovule/fruit development
- We work at the flower level. As we model idealized flowers, and extrapolate to thousands of flowers per crop, relying on averages (e.g. pollinator efficiency) is apropiate.
- We can quickly convert visits per m2 and hour (relevant measure for agronomists) to visits per flower if we know flower density and number of hours flowers are open. e.g.:

  ```
  V_transect_to_flower(V_transect = 50, flw_x_m2 = 400, lifespan = 6)
  ```

  ```
  [1] 75
  ```

- We now eyeball optimal visitation, but we can calculate visits needed to reach 05% of seed set.
- We can use elasticities to see which params have more influence in the outputs.
- We can aproximate the loss to fit the b param autmatically.