Model expressions in Padrino

Sam Levin

Last updated: 2021-03-19

Contents

Overview

Padrino aims to store the mathematical forms of IPMs as text strings. This means that we need to be able to write out the IPM on paper/$LaTeX$/somewhere before we can really know how to enter it. It is important to have a basic understanding of how to write out functional forms for the mean response of a GLM. If you are not already familiar with these, please let speak to either Rob or Sam ASAP so we can get you up to speed!

Models stored in Padrino are rebuilt using ipmr. It will also be helpful to read that project's homepage, as well as the first two articles on simple and general IPMs. These readings will give you a much better understanding of how this package handles different vital rate expressions and kernel formulae. Since Padrino depends on this package to rebuild models, the syntax it uses will look very similar to what should go into Padrino (with a couple exceptions, namely when digitizing probability density functions).

Finally, `ipmr` is designed to generate all model outputs via iteration, rather than generating large kernels and then using linear algebra methods to derive the relevant metrics. Therefore, we don't actually need to digitize the $K(z', z)$ iteration kernel, only the sub-kernels that comprise the model (e.g. $P(z', z), F(z', z), C(z', z)$).

We'll start with simple models, and build complexity as we go through.

Simple models

A simple IPM is an IPM that makes use of 1, and only 1, continuous state variable. It does not include any additional discrete states. These were more common in the early literature, though now more demanding analyses have required authors to create more complicated models. Models that use more than 1 continuous state, and/or include discrete states, are general models, and are detailed below.

An example model - Summary

Below is a shortened description of a simple IPM for *Lonicera maackii* from Levin et al. 2019.[1] x, y correspond to plant size at time t and $t + 1$, respectively. We will use `size_1` and `size_2` in place of x, y when writing out the model in Padrino format (see below).

1. $s(x)$ (survival of plants) was estimated using a Bayesian logistic regression with both linear and quadratic fixed effects.

2. $g(y, x)$ (growth of surviving plants) was estimated using a linear regression of `y ~ x`. We also fit GAMs to our data to see if they produced better predictions. GAMs produced lower AIC scores, but visual inspection of the data for the competitor removal treatment showed that the GAM overfit the data. Thus, to ensure our control and competitor removal treatment IPMs were comparable, we modeled growth using the the linear model. We correct for eviction by adding evicted individuals into the smallest or largest bins after generating the growth kernel.

3. $f_p(x)$ (probability of becoming reproductive) was estimated using a logistic regression of reproductive status on size. We used a logit link and a quasi-binomial error structure to account for overdispersion in our data.

4. $f_s(x)$ (fruits per reproductive plant) was estimated using a generalized linear model with a log link and a Poisson error structure.

5. f_f (seeds per fruit) is a constant based on sampling fruits from 12 individuals.

6. E_p (seedling establishment probability) was taken from a separate experiment in our study that measured new recruits per estimated seed production per plot (Guthrie et al. 2016).

7. $f_d(y)$ (recruit size distribution) was estimated as a Gaussian distribution with mean μ and standard deviation σ from the 17 new recruits that we found in July of 2013.

8. L, U are constants corresponding to the lower and upper bounds of integration. L was defined as the smallest observed size $\times$ 0.9, and U was defined as the largest observed size $\times$ 1.1. We used 500 meshpoints for integration.

9. $n(x, t)$ is the relative frequency of individuals of size x at time t.

The kernels take the following form:

1. $n(y, t + 1) = \int_L^U [P(y, x) + F(y, x)] n(x, t) dx$

2. $P(y, x) = s(x) * g(y, x)$

3. $F(y, x) = E_p * f_p(x) * f_s(x) * f_d(y)$

---

[1]We will only look at the control treatment for now

The vital rates take the following form:[2]

1. $Logit(s(x)) = \alpha_s + \beta_{s,1} * x + \beta_{s,2} * x^2$

    - $\alpha_s = $ -2.831, $\beta_{s,1} = 0.4035$, $\beta_{s,2} = $ -0.000421

2. $g(y, x) = \alpha_g + \beta_g * x$

    - $\alpha_g = 16.884$, $\beta_g = 0.9972$, $sd(g(y,x)) = 32.74774$

3. $Logit(f_p(x)) = \alpha_{f_p} + \beta_{f_p} * x$

    - $\alpha_{f_p} = $ -10.4478, $\beta_{f_p} = 0.0485$

4. $Log(f_s(x)) = \alpha_{f_s} + \beta_{f_s} * x$

    - $\alpha_{f_s} = 3.391$, $\beta_{f_s} = 0.0105$

5. $f_d(y) = Norm(\mu_{f_d} = 3.118, \sigma_{f_d} = 1.215)$

How to digitize this?

Sub-kernels

We'll start with the IPM kernels. Padrino basically uses the mathematical notation from above, but removes the (x, y) portion. So the $P(x, y) = s(x) * g(y, x)$ becomes `"P = s * g"`. Similarly, $F(y, x) = ...$ becomes `"F = e_p * f_p * f_s * f_d"`. We do not need to write out the model iteration from Equation 1. `ipmr` infers the correct form of that, provided we use the right states in `"domain_start"` and `"domain_end"` columns for each kernel formula.

Another important thing to note is that is that it really doesn't matter what you call the kernels, vital rates, and parameters. What matters is that their names match across the various tables in the database. Thus, if an author gives some vital rate a ludicrously long name, feel free to shorten it. The examples below use the same names as the notation above for kernels, vital rates, and parameters, but this is not required!

Finally, because of rather foolhardy design decisions early on, **you must lways separate the left and right hand side of an = with a space.** For example, the `"P = s * g"` from above will not work if it is entered as `"P=s*g"`. On the bright side, it will make the formulae that you enter easier to read![3]

Vital rate expressions

Padrino uses a similar notation as above, except now there are two new elements:

1. We will move all the link functions on the left hand side of each equation to the right hand side, and use the inverse link function instead.

2. We only remove the x, y from the left hand side of the equation.

Survival   The survival function $Logit(s(x)) = \alpha_s + \beta_{s,1} * x + \beta_{s,2} * x^2$ becomes:

+ `` `"s = 1 / (1 + exp(-(alpha_s + beta_s_1 * size_1 + beta_s_2 * size_1 ^ 2)))"` ``

The `1 / (1 + exp(- (...)))` is the inverse logit transformation. This generates probabilities of survival, which is what we want for the IPM. Additionally, notice how the x's were replaced by `"size_1"`'s on the right hand side of the equation. Next, we'll deal with the growth kernel.

---

[2]The notation used here is from the paper. It is not necessarily correct.

[3]I hope to update this to take names from a separate column at some point, rather than using the left hand side of the `=`. However, since it currently works and only requires a bit of care when digitizing, this change isn't particularly high on my priority list.

Growth

In this case, the growth kernel uses a strange notation that doesn't really follow any established convention.[4] However, the written description of the model indicates this is a linear regression with a Gaussian error structure. We would probably do well to re-write this model using a more conventional notation. We can do this like so:

1. $g(y, x) = f_g(y, \mu_g, \sigma_g)$ Where f_g denotes a Gaussian probability density function.

2. $\mu_g = \alpha_g + \beta_g * x$

This format will correspond more closely to how we enter it Padrino. Note that Padrino does not use the R functions like `dnorm`, `dgamma`, etc. Rather, it uses its own shorthand that roughly follows the accepted statistical abbreviations for each probability function family (with some exceptions to avoid name conflicts in R/other languages). The Padrino format for (1) and (2) become:

```
+ `"g = Norm(mu_g, sigma_g)"`
```

```
+ `"mu_g = alpha_g + beta_g * size_1"`
```

Notice that we omit the y from the right hand side of (1). Padrino will infer the correct state variable based on the kernel it appears in. Additionally, in this particular case, their is no link function (or, it's an identity link function which we can ignore), so we don't need to wrap the right hand side of (2) in any thing.

Seed production

The next vital rate we'll go to is seed production: $Log(f_s(x)) = \alpha_{f_s} + \beta_{f_s} * x$. Much like the survival model, we'll need to use an inverse link function on the right hand side. Unlike the survival model, we don't need the inverse logit, but rather, the exponential function (the inverse of the Log).

```
+ `"f_s = exp(alpha_f_s + beta_f_s * size_1)"`
```

Probability of flowering and recruit size distributions.

These are similar to survival probability and the growth distributions. In the spirit of learning, try writing these down on your own and then checking them against the answers given down here.

General models

Above, we dealt with a simple model. It was a simple model because it used only 1 state variable, and had no discrete states. The vast majority of newly published IPMs are not simple models. Complex lifecycles often cannot be described by a single trait value. General IPMs allow for multiple continuous traits, and/or multiple discrete states. In Padrino, little changes with respect to vital rate expressions and parameter values. However, the IPM kernels table will have more entries per model now, as we need more sub-kernels to describe the extra transitions.

Examples of discrete states are things like a seed bank for a plant, or the number of eggs produced by an insect that do not hatch immediately. These are discrete because they do not have any continuous trait that describes their population dynamics, only a single number indicating how many of them there are.

Some IPMs may have multiple continuous state variables. For example, a tree species may use height to describe dynamics of a seedlings below a certain size threshold, and then DBH (diameter at breast height) to describe the dynamics of larger trees.

---

[4]Rest assured, the author of said paper has done much soul searching in the intervening years, and has found a better way of describing these models in subsequent publications.

Below, we'll go through another example from Baer & Maron (2018). We'll look at a single transition year from a single population first, and then use the rest of the data in the next example to demonstrate the suffix syntax that Padrino uses. You can find the main text here and the appendix containing the necessary information here.

Where to begin

Assuming we've already entered most of the Metadata table, the first step is figuring out where to get the rest of it. Browsing through the paper, it looks like section 2.4 is likely to contain the information on the actual IPM. The end of the first paragraph contains key information on the number of kernels:

> "Our IPMs contained a continuous stage describing size based vital rates of juvenile and reproductive adult plants and two discrete stages in which constants described the vital rates of seedlings (whose size was not recorded) and seeds within the seed bank. The continuous stage of our IPMs was structured according to logtransformed basal rosette area (hereafter, size) and integrated across a range of sizes from the minimum size to 1.2 times the maximum size of any individual recorded in a population throughout all years of the study."

Ok, so we have 3 different states that we'll need to track: size of non-seedling plants, the number of seedlings, and the number of seeds in the seedbank. Next, we need to find out how they move around within and between those states. It becomes apparent from reading on that we won't find that information in the main text. However, the first sentence in the second column on page 5 indicates that we might be able to find it in Appendix S1. Let's open that up!

Digitizing the kernel formulae

Low and behold, the mathematical form of the IPM iteration is right here! Hooray! Let's start with the seed bank terms. They've written it out as:

$$B_{t+1} = B_t * p_{sbanksurv} * (1 - p_{sbankgerm}) + \int_L^U p_{flower}(x) * f_{seeds}(x) * (1 - p_{germ})n_t(x)dx$$

This expression is comprised of two sub-kernels, which we can recognize by the `+` that occurs about halfway through, and the fact that one part is multiplied by a discrete state (B_t) and another part is multiplied by a continuous state ($n_t(x)$). Thus, we need to split this single expression into two lines in the `IpmKernels` table. Bearing in mind that we do not include the actual state in these expressions (i.e. B_t and $n_t(x)$), these will take the form (column names now separated by commas):

1. formula = `"SbStay = sb_surv * (1 - sb_germ)"`, model_family = `"DD"`, domain_start = `"seedbank"`, domain_end = `"seedbank"`

2. `"SbGo = p_fl * n_seeds * (1 - p_germ) * d_size"`, model_family = `"CD"`, domain_start = `"size"`, domain_end = `"seedbank"`

One critical aspect to note is that in this case, we've added the `"d_size"` variable to expression 2. `ipmr` requires us to specify the integration for general IPMs, whereas it will automatically infer it for simple ones. In all cases, if you encounter an integral, this `"d_size"` will correspond to the dx in the integral. We just need to make sure we choose the state variable.

Additionally, we now have different values for model_family, domain_start, and domain_end.

We can now move onto the second expression in the IPM iteration:

$$S_{t+1} = p_{sbankgerm} * B_t + \int_L^U p_{flower}(x) * f_{seeds}(x) * p_{germ}n_t(x)dx$$

Again, we have two sub-kernels defined in this portion of the iteration. The first moves seeds out of the seedbank to create seedlings, and the other creates seedlings from adult plants that are reproductive. We write them as follows:

1. formula = "SbSdl = p_sb_germ", model_family = "DD", domain_start = "seedbank", domain_end = "seedling"

2. formula = "goSdl = p_fl * n_seeds * p_germ * d_size", model_family = "CD", domain_start = "size", domain_end = "seedling"

The final part of the IPM iteration generates plants with some size distribution. It has the following form:

$$K = p_{seedlingsurv} * f_{recrsize}(x') * S_t + \int_L^U s(x) * g(x'|x)n_t(x)dx$$

Try writing out the formula, model_family, domain_start, and domain_end on your own. You can check your answers here.[5]

Vital rate expressions

Most of the vital rate expressions are only described qualitatively, rather than explicitly written out. Additionally, they are mostly written out in the main text, specifically in the second and third paragraphs of section 2.4. We'll start by parsing the following section:

> "Within the continuous portion of the IPMs, logistic regressions with a logit link described size-based binomial survival probability. Linear models described size-based growth rates and variance around this relationship. Log-transformation of plant size eliminated the need for a size-based estimate of the variance around the size– growth relationship for most models, as verified by a Breusch–Pagan test performed on each model for this relationship (bptest, lmtest package, Zeileis & Hothorn, 2002). In these cases, variance was described according to the residual standard deviation. For size-growth models that remained heteroscedastic following log-transformation of plant size, we described variance in growth as an exponential function of log-transformed plant size when constructing growth models: $\sigma(size) = sqrt(\sigma^{2[2*constant*size]})$, where the constant and σ^2 were calculated within IPMpack"

Suffix syntax

Parameter resampled models

Age × size models

Answers to exercises

Simple model Pr(flowering) and recruit size.

1. $Logit(f_p(x)) = \alpha_{f_p} + \beta_{f_p} * x$ becomes "f_p = 1 / (1 + exp(-(alpha_f_p + beta_f_p * size_1)))"

2. $f_d(y) = Norm(\mu_{f_d}, \sigma_{f_d})$ becomes "f_d = Norm(mu_f_d, sigma_f_d)"

---

[5]Bonus: the notation in this particular equation is somewhat unconventional. Can you re-write it to specify the continuous portion of the population state at $t + 1$ (i.e. $n(x', t + 1) = ...$?

General IPM kernel formulae

$$K = p_{seedlingsurv} * f_{recrsize}(x') * S_t + \int_L^U s(x) * g(x'|x)n_t(x)dx$$

1. formula = "`SdlOut = p_sdl_surv * sdl_size * d_size`", model_family = "`DC`", domain_start = "`seedling`", domain_end = "`size`"

2. formula = "`P = s * g * d_size`",

   - Bonus: $n(x', t+1) = p_{seedlingsurv} * S_t * \int_L^U f_{recrsize}(x')dx' + \int_L^U s(x) * g(x'|x)n_t(x)dx$