1 Send Population Model

From Simulation, Modeling and Analysis:

> The state of a system is defined to be the collection of variables necessary to describe a system at a particular point in time, relative to the objective of a study.

and

> A system is defined to be the collection of entities that act and interact towards the accomplishment of some goal.

In the Send Model the collection of entities that make up the system are drawn from $S \times N \times AY$ where $S = \{MMSIB, ISS, ISSR, \ldots, MU\}$, $N = \{CL, \ldots, NONSEND\}$, $AY = \{4, \ldots, 24\}$.

Each entity has a single state variable t which represents its total population.

The points in time for which we can examine an entity's state is per calendar year.

The Send simulation uses the population of each of the system entities (population state) for one calendar year to predict the population state of the next calendar year.

By visualising the mathematical structures that govern population state we can gain an intuitive understanding of how the model works.

In Fig.1 below t_{s_x,n_y,ay_z} represents the total population of some setting s_x, need n_y and academic year ay_z.Where x, y, z are any integer index into the respective ordered sets: S, N, AY.

1.1 Markov Chain

This section and the next are a detour into how a the Send model would look if it were implemented using just Markov Chains. If we understand this we understand why we can see the Send Model is" 'Markov like'".

The initial state of our Markov Chain, Y_0 is constructed from the historic data provided by the client (we simply count the number of pupils that appear e.g for a single state index, (s_1, n_1, ay_1), how many rows in our input data match?

In Fig.1 Y_{n+1} is the next state of our Markov Chain.

How we transition from Y_n to Y_{n+1} is the subject of the next section. However, it's worth noting now that the dotted arrows on the diagram represent how the t's of one academic year transition into the next academic year, in the following calendar year.

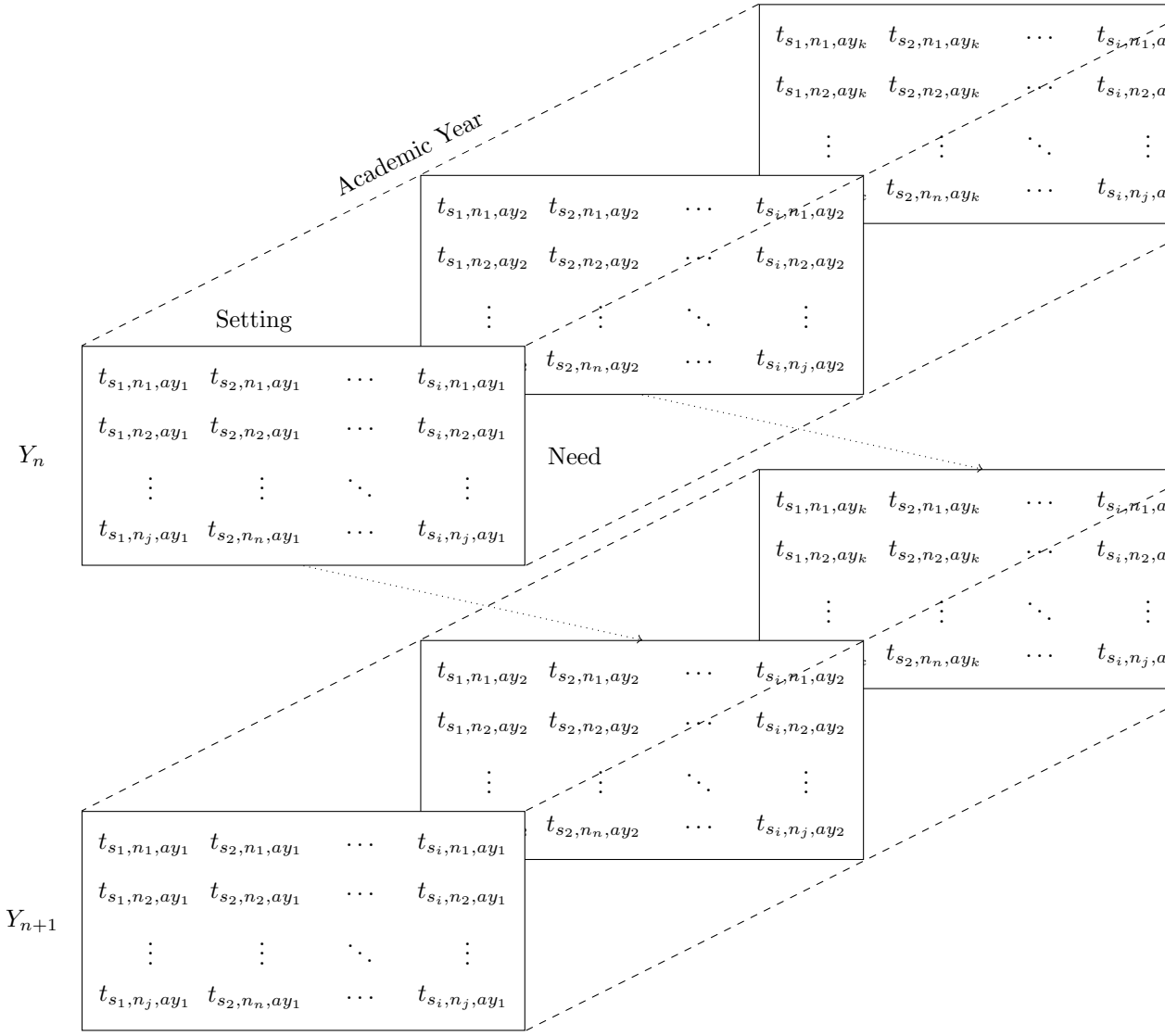

Figure 1: State Transitions in the Send Markov Chain

The complete state of the model is represented as $S = \{Y_0, \ldots, Y_n\}$ where n is the number of years we run the simulation for.

1.2 Markov Transition Matrix

This section can be skipped - it explores what Markov Transition Matrix would look like if the model used it.

In mathematics a Markov Transition Matrix is formally a matrix containing the probability of transitioning between Markov states. *Do not confuse the Markov Transition (Probability) Matrix with the Send Model's Transitions data file.* Conceptually, for the Send Model, it can be thought of as telling us what percentage of t_{s_x,n_y,ay_z} will move to another state for each of the valid (allowed) states that t_{s_x,n_y,ay_z} can move to. Sometimes this is referred to as *rates*.

By looking at the historic data we could work out the expectation of each of our transitions and build this matrix. We visualise what this looks like in Fig.2.

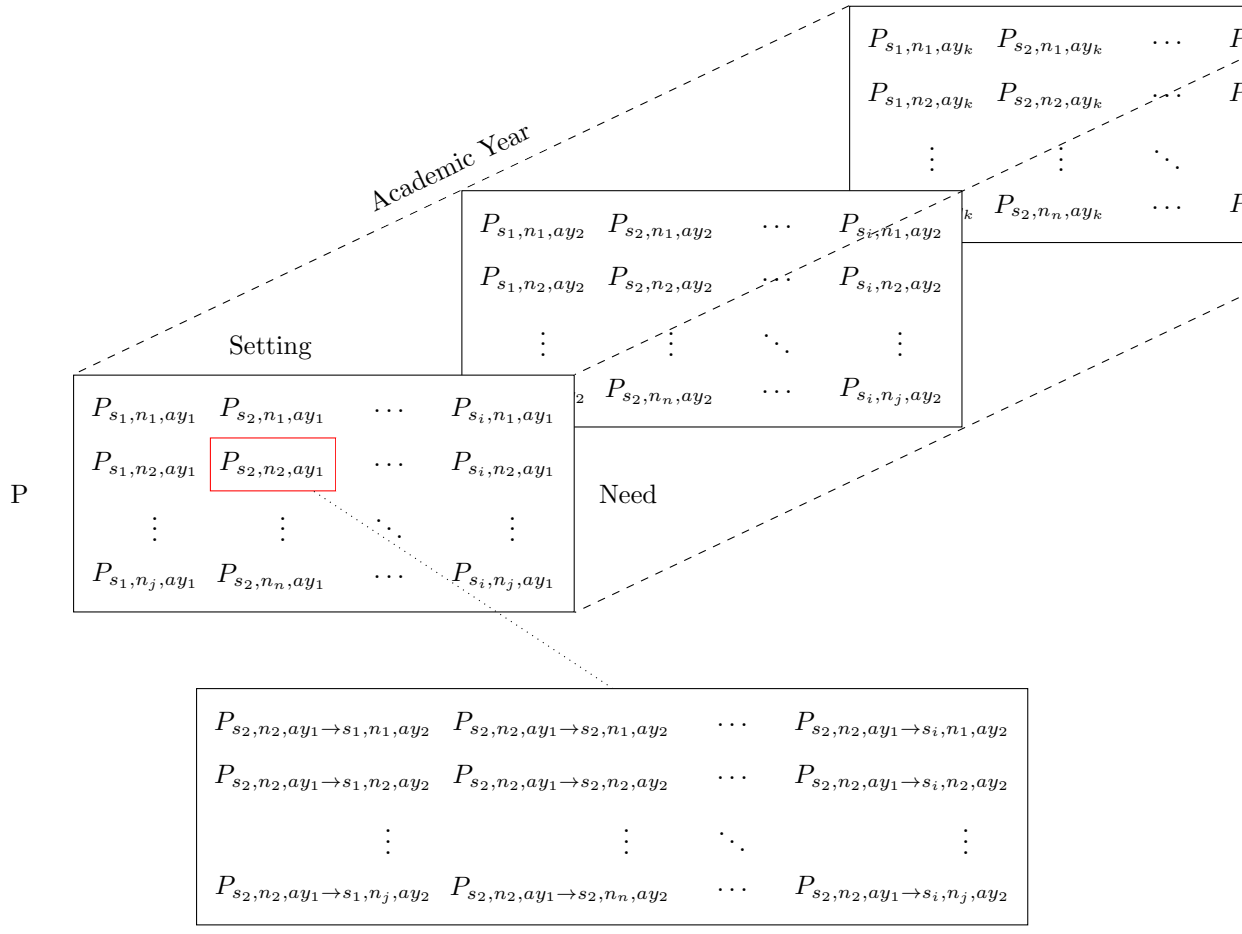

Figure 2: Markov Transition Matrix (many nested probability matrices)

In the Fig.2 every P_{s_i,n_j,ak_1} is a probability matrix that represents the probabilities of transitioning to any other state in the next academic year (see the diagram's cut-out for an example.)

Armed with P, this rather complicated Markov Transition Matrix, we can apply it to the initial state

$Y_1 = Y_0 \cdot P$

In fact at this point as P is time independent we can work the populations in year n by

$Y_n = Y_0 \cdot P^n$

Of note is that this equation requires no tracking of t's we can simply use the above formula to work out Y_n. BUT we don't do this! Why?

1.2.1 Send is not MC

The send model actually sides step the creation of a Markov Transition Matrix and takes an algorithmic approach to calculating Y_{n+1}'s population.

This is done as some of the data is perhaps too sparse, but also the algorithmic approach allows us to introduce priors and scenarios rather more easily than modifying P directly.

Further, we also use more than the just the initial state, in our algorithmic predictions. The model incorporates external population data for each predicted calendar year.

As we will come to see the algorithmic approach to working out the t's in Y_n introduces stochastic processes. Thus we introduce the need to use Monte Carlo methods to ensure confidence in our calculations.

The model can be thought of as a simulation in with a simulation clock that ticks once a calendar year. On clock tick all the entities have there new population calculated: based on there existing state, pre-calculated historic probability distributions and projected population.

1.3 Send Model's Population Prediction Algorithm

The algorithm can be simplified to:

$Y_{n+1} = T(Y_n)$

Y_n represents calendar year n's population state, T the function that calculates the next years population state from it. The simplification of T means that some things that are variable are closed over i.e, the external ONS population data is not a variable, nor are the pre-calculated probability distributions.

Let us consider first a single entity's population t_{s_x,n_y,ay_z}. How will T operate on this?

1.3.1 Movers, Non-Movers, Leavers and Joiners

The model introduces the concepts of movers(m), leavers(l), and joiners(j).

Let m_{s_x,n_y,ay_z} be defined as all the pupils that move **into** a particular entity from another entity.

Let n_{s_x,n_y,ay_z} be defined as all the pupils that **don't** move out of a particular entity (non-movers or stayers).

Let l_{s_x,n_y,ay_z} be defined as all the pupils that **leave** a particular entity and out of the Send programme completely.

Let j_{s_x,n_y,ay_z} be defined as all the pupils that join **into** a particular entity from the outside populace.

So $T(Y_n)_{s_x,n_y,ay_z} = m_{s_x,n_y,ay_z} + n_{s_x,n_y,ay_z} - l_{s_x,n_y,ay_z} + j_{s_x,n_y,ay_z}$

Reading this equation aloud makes obvious intuitive sense. The question now becomes how do we calculate each of the components of this function?

In all the following sections except where noted it is assumed that the underlying processes governing leavers, joiners, and movers are independent and identically distributed meaning that we can combine historic calendar years data as one.

1.3.2 Leavers

From our historic data we can work out the percentages over the calendar years of how many pupils leave a particular entity, (s_1, n_1, ay_1). Averaging this sets our expectation e.g. for (s_1, n_1, ay_1) we may expect 40% of the population to leave.

For the purposes of our simulation we do not want every run to have exactly 40% of the population leave but rather have some variance in the size. How these variances in the population propagate through the simulation lets us see confidence bounds when we come to aggregate the results of many runs of the model.

A first pass at introducing variance is to use the binomial distribution to alter the population that leaves per run.

Continuing our example given $t_{s_1,n_1,ay_1} = 100$ and an expectation of 40% then a sample of ten values looks like so:

```
(sample 10 (binomial {:n 100 :p 0.4}))
(46 42 31 45 43 44 48 41 41 40)
```

We can further refine the introduction of variance by noting that the expectation percentage of leavers is stronger or weaker (more or less confident in) depending on the amount of data we have. Intuitively if we have lots of observations around the number of leavers then we can be much more confident in our expectation value. If we have very few observations then we are much less confident it's the correct value.

The beta distribution can be used to express the variance in our expected leaver percentage. The parameters for the distribution α, β are simply the observed leavers and non-leavers respectively. Drawing from the distribution is equivalent to working out the percentage expectation (obviously slightly changed as it's purpose is to introduce variance in this value). For low values of observations there will be a wide range in the values of the expected leavers and for high values of observations there will be a much narrower range of expected values.

If we use the results of the Beta distribution as the probability parameter of the Binomial distribution (composition) we form what's called a hierarchical model. This chaining of distributions increases the size of variance in our model runs.

Thus we can write:

$$l_{s_x,n_y,ay_z} = \text{Binomial}(t_{s_x,n_y,ay_z}, \text{Beta}(\alpha^l_{s_x,n_y,ay_z}, \beta^l_{s_x,n_y,ay_z}))$$

where $\alpha^l_{s_x,n_y,ay_z}$ represents the count of leavers over all historic calendar years for the entity (s_x, n_y, ay_z). Similarly $\beta^l_{s_x,n_y,ay_z}$ represents the count of all non leavers for the same entity over all historic calendar years. (t_{s_x,n_y,ay_z} represents the populatio of the entity.

The algorithm described here is a high level overview of the implementation by the Send code base.

1.3.3 Joiners

The calculation for working out an entity's joiners is split into two parts: first the expected rate of joiners is calculated per academic year, then this additional population is assigned a need and setting.

The sum of the joiners for a single academic year over all historic calendar years is first calculated.

We work out the number of non-joiners by taking the joiners away from the predicted number of pupils for an AY (e.g from ONS data). This additional population data source (in addition to the entity state that is) is typically only available at an AY level. Our desire to use it means that we can't work at the normal s_x, n_y, ay_z level.

Like for the leavers we wish to have a variance in this value based on our confidence of the data. Like before, we can apply the beta distribution to

do this. However, in our code base we divide through the sum of the joiners and non-joiners by observed calendar years. This has the effect of increasing variance. It is in direct contrast to the other probability distribution calculations that assume iid and simply sum these values.

We once more use the binomial distribution to provide variance in our population values once we have an expected probability from the beta distribution. Thus:

$$j_{ay_z} = \text{Binomial}(p_{ay_z}, \text{Beta}(\frac{\alpha^j_{ay_z}}{cy_{obs}}, \frac{p_{ay_z} - \alpha^j_{ay_z}}{cy_{obs}})$$

where $\alpha^j_{ay_z}$ are the joiners for ay_z (from the historic calendar years), cy_{obs} is the number of observed calendar years, and p_{ay_z} is the projected population from our additional population source.

The second part of the calculation is to distribute the new joiners to appropriate needs and settings. We count for each academic year exactly how many joiners there are from non-send to each of the need-settings. The proportions of which tell us how to distribute the joiners. Note: this time we don't divide through by the academic years. Armed with these proportions we can calculate the expectation as a percentage for each e.g 10% to (s_x, n_y, ay_z) and 90% to (s_j, n_k, ay_l).

Again, rather then a direct calculation of the population to be assigned to each entity we introduce variance by using the multinomial distribution. It performs exactly the same job as the binomial except over multiple possible outcomes.

In a now familiar pattern, we also wish to introduce variance in the actual percentages used to calculate the assignment. Like before, we want high confidence in values we have lots of observed data for and low confidence for few observations. The Dirichlet distribution does this for multivariates in exactly the same way beta distribution does for two. We combine the two to produce another hierarchical model.

$$\begin{aligned}&\text{Multinomial}(j_{ay_z},\\ &\qquad\text{Dirichlet}(\alpha^j_{s_x,n_y,ay_z}, \ldots, \alpha^{j''}_{s_{x''},n_{y''},ay_{z''}}))\end{aligned}$$

Here the α^js represent the total number of transitions from *nonsend* to s_x, n_y, ay_z.

The results of the multinomial function is however a vector of the counts of the joiners for each of the possible entities. Thus for any particular entity of interest we need to take the appropriate index.

$$j_{s_x,n_y,ay_z} = \Big[\text{Multinomial}(j_{ay_z}, \text{Dirichlet}(\alpha^{j}_{s_x,n_y,ay_z}, \alpha^{j'}_{s_{x'},n_{y'},ay_{z'}}, \ldots, \alpha^{j''}_{s_{x''},n_{y''},ay_{z''}}))\Big]_{s_x,n_y,ay_z}$$

We can write this out in full for single entity by expanding j_{ay_z}.

$$j_{s_x,n_y,ay_z} = \Big[\text{Multinomial}(\text{Binomial}(p_{ay_z}, \text{Beta}(\frac{\alpha^{j}_{ay_z}}{cy_{obs}}, \frac{p_{ay_z} - \alpha^{j}_{ay_z}}{cy_{obs}}), \text{Dirichlet}(\alpha^{j}_{s_x,n_y,ay_z}, \ldots, \alpha^{j''}_{s_{x''},n_{y''},ay_{z''}}))\Big]_{s_x,n_y,ay_z}$$

Interestingly, we can immediately see that the number of joiners to an entity is not directly related to that entity's population state, rather it's relation is to the projected population for an academic year and the non-send to entity transitions in the observed data.

1.3.4 Movers and Non Movers

The base form of the calculations should be by now familiar.

$$m^{partial}_{s_x,n_y,ay_z} = \text{Multinomial}(\text{Binomial}(t_{s_x,n_y,ay_z} - l_{s_x,n_y,ay_z}, \text{Beta}(\alpha^{m}_{s_x,n_y,ay_z}, \beta^{m}_{s_x,n_y,ay_z})), \text{Dirichlet}(\alpha^{m'}_{s'_x,n'_y,ay'_z}, \ldots, \alpha^{m''}_{s_{x''},n_{y''},ay_{z''}}))$$

The first thing to note is that this a partial result - which we will come to later.

Like leavers the relationship is based on the current entity's population count (not the academic year's population count). However it is slightly modified in that we take the previously calculated leavers away for this entity (as we know they move to non-send).

The Beta params α^m and β^m are the counts of the transitions out-of-s_x, n_y, ay_z-and-into-some-other-setting and the transitions to-itself respectively. Unlike joiners they are not divided through by observed calendar years.

The Dirichlet params $\alpha^{m'}$'s are the counts from the observed data of our entity s_x, n_y, ay_z to the other entities it can move to s'_x, n'_y, ay'_z (represented by ever increasing dashes). As such the vector that is the result of the multinomial has no count of any movers to our entity s_x, n_y, ay_z. Instead we need to calculate $m^{partial}$ for every other entity and take from each generated vector the index

that represents the population that moves to m_{s_x,n_y,ay_z}. The sum of these is the total movers into our entity.

$$m_{s_x,n_y,ay_z} = \sum_{j\neq x,k\neq y,l\neq z} \Big[\text{Multinomial}(\text{Binomial}(t_{s_j,n_k,ay_l} - l_{s_j,n_k,ay_l}, \text{Beta}(\alpha^m_{s_j,n_k,ay_l}, \beta^m_{s_j,n_k,ay_l})), \\ \text{Dirichlet}(\alpha^{m'}_{s'_j,n'_k,ay'_l}, \ldots, \alpha^{m''}_{s_{j''},n_{k''},ay_{l''}}))\Big]_{s_x,n_y,ay_z}$$

where j, k, l belong to S, N, AY respectively.

As we know that non-movers for an entity stay within the entity we can just use the beta binomial to calculate these.

$$\begin{aligned} n_{s_x,n_y,ay_z} =& t_{s_x,n_y,ay_z} - l_{s_x,n_y,ay_z} \\ & - \text{Binomial}(t_{s_x,n_y,ay_z} - l_{s_x,n_y,ay_z}, \text{Beta}(\alpha^m_{s_x,n_y,ay_z}, \beta^m_{s_x,n_y,ay_z})) \end{aligned}$$

For consistency of the model that Bionomial component of the above equation should be the same result as when it was used to calculate the movers.

1.3.5 The Full Algorithm

Calculate next years population T for an entity given the current years state.

$Y_{n+1 for s_x,n_y,ay_z} = T(Y_n)_{s_x,n_y,ay_z} = m_{s_x,n_y,ay_z} + n_{s_x,n_y,ay_z} - l_{s_x,n_y,ay_z} + j_{s_x,n_y,ay_z}$

Where

$$m_{s_x,n_y,ay_z} = \sum_{j\neq x,k\neq y,l\neq z} \Big[\text{Multinomial}(\text{Binomial}(t_{s_j,n_k,ay_l} - l_{s_j,n_k,ay_l}, \text{Beta}(\alpha^m_{s_j,n_k,ay_l}, \beta^m_{s_j,n_k,ay_l})), \\ \text{Dirichlet}(\alpha^{m'}_{s'_j,n'_k,ay'_l}, \ldots, \alpha^{m''}_{s_{j''},n_{k''},ay_{l''}}))\Big]_{s_x,n_y,ay_z}$$

$$n_{s_x,n_y,ay_z} = t_{s_x,n_y,ay_z} - l_{s_x,n_y,ay_z} - \text{Binomial}(t_{s_x,n_y,ay_z} - l_{s_x,n_y,ay_z}, \quad \text{Beta}(\alpha^m_{s_x,n_y,ay_z}, \beta^m_{s_x,n_y,ay_z}))$$

$$l_{s_x,n_y,ay_z} = \text{Binomial}(t_{s_x,n_y,ay_z}, \text{Beta}(\alpha^l_{s_x,n_y,ay_z}, \beta^l_{s_x,n_y,ay_z}))$$

$$j_{s_x,n_y,ay_z} = \Big[\text{Multinomial}(\text{Binomial}(p_{ay_z}, \text{Beta}(\frac{\alpha^j_{ay_z}}{cy_{obs}}, \frac{p_{ay_z} - \alpha^j_{ay_z}}{cy_{obs}}), \\ \text{Dirichlet}(\alpha^j_{s_x,n_y,ay_z}, \ldots, \alpha^{j''}_{s_{x''},n_{y''},ay_{z''}}))\Big]_{s_x,n_y,ay_z}$$

From the equations we see that the various parameters beta and Dirichlet parameters are closed over by this alogorithm - they are calculated once ahead of time as described in each subsection.

The entire current state Y_n is required to work out a single entity's future population: this is because we use the entire AY population in the joiner calculation and we need to sum all movers into our entity from all the other entities.

1.4 A Case Study

We go through the calculation of an entity's next state to show how the algorithm works and to embed the meaning of the myriad of parameters.

Let's us consider need a, setting b, in academic year 1, the calendar year of 2018 - this entity at this time has a population of 100. The algorithm will give us this entity's population for the next calendar year 2019.

$Y_{2019 for s_a,n_b,ay_1} = T(Y_{2018})_{s_a,n_b,ay_1} = m_{s_a,n_b,ay_1} + n_{s_a,n_b,ay_1} - l_{s_a,n_b,ay_1} + j_{s_a,n_b,ay_1}$

and

$t_{s_a,n_b,ay_1} = 100$

First-off the leavers as it's the simplest:

$$l_{s_a,n_b,ay_1} = \text{Binomial}(t_{s_a,n_b,ay_1}, \text{Beta}(\alpha^l_{s_a,n_b,ay_1}, \beta^l_{s_a,n_b,ay_1}))$$

Let's say we already know the expected rate of leavers is 40% then

$t_{s_a,n_b,ay_1} \times 40\% = 100 \times 40 = 40 pupils$

We want variance in that so we use the binomial distribution to draw some samples

$$\text{Binomial}(t_{s_a,n_b,ay_1}, 0.4) = \text{Binomial}(100, 0.4) = 38$$

Now we also want variance in the 0.4 value, so we use the Beta distribution. Assume that we have historic data for last 2 years that has 60 and 20 leavers, and 110 and 90 non-leavers respectively.

$$\text{Beta}(\alpha^l_{s_a,n_b,ay_1}, \beta^l_{s_a,n_b,ay_1}) = \text{Beta}(60 + 20, 110 + 90) = \text{Beta}(80, 200) = 0.38$$

Combining this all together

$$l_{s_a,n_b,ay_1} = \text{Binomial}(100, \text{Beta}(80, 200)) = 43$$

Next let's look at joiners.

$$j_{s_a,n_b,ay_1} = \Bigg[\text{Multinomial}(\text{Binomial}(p_{ay_1}, \text{Beta}(\frac{\alpha^j_{ay_1}}{cy_{obs}}, \frac{p_{ay_1} - \alpha^j_{ay_1}}{cy_{obs}}), \\ \text{Dirichlet}(\alpha^j_{s_x,n_y,ay_z}, \ldots, \alpha^{j''}_{s_{x''},n_{y''},ay_{z''}}))\Bigg]_{s_a,n_b,ay_1}$$

Let's assume our external population p_{ay_1} is predicted to be a 1000 for ay_1 in 2019.

For our two years of historic data let's assume the predicted populations are 900 and 1100.

Let's also say that the number of joiners in our historic data is 30 and 20 over all needs and settings for ay_1.

A first pass at expectation based on this data is:

$\frac{30+20}{900+1100} = 0.025$

Like before we need would like some variance in this value, so we use the beta distribution again.

$$\text{Beta}(\frac{\alpha^j_{ay_1}}{cy_{obs}}, \frac{p_{ay_1} - \alpha^j_{ay_1}}{cy_{obs}}) = \text{Beta}(\frac{20+30}{2}, \frac{900+1100-20-30}{2}) = \text{Beta}(25, 975) = 0.02501$$

note: ignore the division through by calendar years for now (it doesn't effect expectation only the variance)

We once more use the binomial to get a varied value from the population and the expectation.

$\text{Binomial}(p_{ay_1}, 0.02501) = \text{Binomial}(1000, 0.02501) = 25$

Armed with the fact we have 25 joiners for academic year 1 in 2019 our equation reduces to:

$$j_{s_a,n_b,ay_1} = \Bigg[\text{Multinomial}(25, \text{Dirichlet}(\alpha^j_{s_a,n_b,ay_1}, \ldots, \alpha^{j''}_{s_{x''},n_{y''},ay_1}))\Bigg]_{s_a,n_b,ay_1}$$

This part deals with assigning those 25 pupils to specific entities. Assuming our historic data for academic year one has one other valid setting need c and setting d - what are the proportions of the population between the two entities?

Let's say ab1 has populations of 20 and 40 over the two years and cd1 has populations of 60 an 80. Then we can say 30% of the population is in ab1 and 70% of the population is in cd1.

We can use these percentages to distribute the 25 joiners appropriately. Like before we'd like to vary this and the dirichlet performs the same job as the beta in this regard. So,

$$\begin{aligned}\text{Dirichlet}(\alpha^{j}_{s_a,n_b,ay_1},\ldots,\alpha^{j''}_{s_{x''},n_{y''},ay_1}) &= \text{Dirichlet}(20+40_{s_a,n_b,ay_1},60+80_{s_c,n_d,ay_1})\\ &= \text{Dirichlet}(60_{s_a,n_b,ay_1},140_{s_c,n_d,ay_1})\\ &= [0.29_{s_a,n_b,ay_1},0.71_{s_c,n_d,ay_1}]\end{aligned}$$

Again rather than simply calculate the distribution based on these percentages we introduce variance by using the multinomial to vary the results (this plays the same role as the binomial from before). Thus

$$\text{Multinomial}(25,[0.29_{s_a,n_b,ay_1},0.71_{s_c,n_d,ay_1}]) = [7_{s_a,n_b,ay_1},17_{s_c,n_d,ay_1}]$$

The result is a vector and as we calculating for only s_a, n_b, ay_1 the last thing we do is pull it's index out (the square brackets in the algorithm and it's index represent that). Thus we have 7 joiners in this case.

2 Other Things To Do

- Priors section for leavers, joiners, movers
- Scenarios
- Valid state implications
- External population control
- The real simulation tick mechanism