# Robustness

### Assumptions

In building our model, we made various assumptions about our functions, especially with respect to our cost and utility functions. As a result, we used some arbitrary functions to model utility. Ultimately, these functions contributed to our objective function, which meant that they were a significant factor in our model.

For the sake of robustness, in this section, we would like to address some of these concerns, to see if using alternative modelling approaches would have affected our results. Our assumptions and their corresponding check for robustness are listed below:

1. **Equal Distance in Rankings**: We assume that utility rankings are equidistant, hence using a linear function to assign a utility multiplier.

Robustness Check: we adjust the utility multiplier of each factor by ±30% with respect to the others – 3 factors are tested at 3 levels.

1. **Module Preference 0.25 Damping Factor**: We utilized an exponential function to convert module preferences into utility, using the formula . However, 0.25 is an arbitrary decision.

Robustness Check: we adjust the dampening factor by ±30% with respect to other factors – one factor tested at 3 levels.

1. **Equal Cost at Every Timeslot**: we assumed that every time slot has an equivalent cost, i.e. 0800-0900 holds the same cost to a student has 1000-1100. However, this assumption likely does not hold; we expect students to very much prefer a 10am class over an 8am one.

Robustness Check: we added alternative timeslot cost functions (linear, quadratic) to our model – one factor tested at 3 levels.

1. **Equal Preference for Early and Late**: we assumed that students disliked early days as much as late days; students have differing preferences over early and late hours.

No robustness testing was conducted here. Instead, we recognized this as a feature that can be trivially implemented in the future.

1. **Equal Preference for Days of Week**: we assumed that every day meant the same for students; students may prefer an extended weekend, or a mid-week break.

No robustness testing was conducted here. Instead, we recognized this as a feature that can be trivially implemented in the future.

Based on these robustness testing decisions, we identified 5 factors to test, each with 3 levels (+30%, base, -30%). This gives us a total of 35 = 243 cases for each of our test cases. Our results with these test cases will be discussed below.

### Results

Using the above-mentioned approach, we ran 243 different optimization models on our test case as described in chapter **X**. Surprisingly, from these test cases, we saw that there was absolutely no change in the modules chosen. In the table below, the left column represents the module-sectional to take, and the right column represents the percentage of times this slot was chosen in the 243 models we ran.

|  |  |
| --- | --- |
| ACC1006 -J2 | 1 |
| BSP1004-A2 | 1 |
| BSP1005-D1 | 1 |
| DSC1007-B10 | 1 |
| FIN2004-J08 | 1 |

While these results were satisfactory, they were not fully indicative, as there might be a simple solution to this problem that did not have a need to balance of different priorities and weights. Nevertheless, it suggests that our model is sufficiently robust to provide solutions to an average person’s module choices, which did not involve complex choice decisions.

We moved on to use a complex test case for stress testing: we picked the modules with the most combinations of sections, and ranked 8am classes very highly. This way, the modules that had a large utility for being top ranked also had the highest cost for being early in the day. At a sensitivity of 30%, we obtained the following results.

|  |  |
| --- | --- |
| MNO1001-AA1 | 0.926 |
| MNO1001-CC3 | 0.074 |
| ACC1002-V1 | 1.000 |
| BSP1005-D1 | 0.938 |
| DSC1007-B2 | 0.061 |
| DSC1007-B10 | 0.926 |
| MK1003-AA1 | 1.000 |
| DSC2006-F1 | 1.000 |

Despite some variation, we are rather satisfied with the results. With a variation of >10% for all modules, it goes to show that our timetable builder is still relatively robustness, especially considering this was an extreme test case. Furthermore, these variations could also be used to present alternatives to the users of our model, should there be any. Overall, with these results, we accepted our model as acceptably robust and made no further modifications.