HW5 ETM 540

Risto Rushford

2021-02-26

Exercise 5.8: A DEA Application

Background

We are given the task of finding an application with readily available data with which we can perform our own Data Envelopment Analysis using the TRA package in R. The application that I have chosen to use comes from the SpaceFund Launch Database

The data set used in this exercise has a subset of the data pulled form the Launch Database, with only companies that are at or near operation, with publicly available figures on total funding, and is restricted to companies competing in the small launch subsegment of the Space Launch Industry.

Application Description

As previously mentioned, the application of interest here pertains to small space launch providers and the relationship between funding and launch capabilities. This is of special interest because, as Tory Bruno, the CEO of United Launch Alliance, has remarked recently, the space launch market is "overheated," with too much funding being poured into risky ventures while the space launch market's customer base remains nearly static in size.

This is especially true in the Small Launch Provider segment of the market, with few currently operating commercially yet several dozen companies world-wide are looking to begin their own commercial operations in the very near future (some of them this year). Given this information, let's evaluate the 7 best funded small launch providers with publicly available data. Note that of these companies, only Rocket Lab currently has ongoing operations and that for the rest of the companies, ultimate funding levels and operating costs are likely to vary significantly until these companies begin commercial launch operations.

Data Preparation

First we'll upload the data and have a look. Note that the information available from each company for our DEA is total funding ($M), cost per launch ($M), price per kilogram, and total payload capacity.

Uploading the Data

```
launch_providers_2020 <- read_csv("launch_providers_2020.csv",
                                  col_types = cols(), col_names = TRUE)

launch2020 <- launch_providers_2020 [1:7,]
```

Table 1: Data for Space Launch Providers in 2020

Alphas	Commercial Launch Providers	Country	Launch Cost ($M)	Price ($/kg)	Payload (kg)	Funding ($M)
A	Rocket Lab	USA	4.9	16333	300	257.31
B	Astra Space	USA	2.5	12255	204	100.00
C	iSpace	China	5.0	16667	300	275.00
D	Firefly Aerospace	USA	15.0	15000	1000	23.16
E	Relativity Space	USA	10.0	8000	1250	684.54
F	ABL Space Systems	USA	12.0	8889	1350	49.00
G	One Space	China	3.2	16000	200	116.00

Formatting the Data

Now we've uploaded our data and confirmed that it is useful and complete, let's prepare it to be used in our model. Note the *Alphas* column, because I will use these to label the companies in graphs later.

```
RTS<-""

xdata <- as.matrix(launch2020[,7])
rownames(xdata)<-as.matrix(launch2020[,2])

ydata <- as.matrix(launch2020 [,4:5])
rownames(ydata)<-as.matrix(launch2020[,2])

Xnames <- colnames(xdata)
Ynames <- colnames(ydata)
DMUnames <- list(as.matrix(launch2020[,2]))

dimnames(xdata)              <- c(DMUnames,Xnames)
colnames(ydata)              <- Ynames

ND <- nrow(xdata)  # Number of DMUs (Launch Providers)
NX <- ncol(xdata)  # Number of inputs (just 1 in this case)
NY <- ncol(ydata)  # Number of outputs

res.efficiency <- matrix(rep(-1.0, ND), nrow=ND, ncol=1)
res.lambda     <- matrix(rep(-1.0, ND^2), nrow=ND,ncol=ND)
dimnames(res.efficiency) <- c(DMUnames,paste(RTS, '- IO'))
dimnames(res.lambda)     <- c(DMUnames,DMUnames)

#Define printable names to be used as appropriate
```

```r
ynames_printable <- c("Launch Cost ($M)",
                      "Price ($/kg)")
xnames_printable <- c("Funding ($M)")
DMUnames_printable <- as.matrix(launch2020[,2])
Alphas <- as.matrix(launch2020[,1])
```

Before I proceed, let's preview the formatted data to make sure that it's still correct. Notice that I dropped the *Alphas*, *country*, and *Payload* columns, and shifted the *Funding* column to where the country column was in the original dataset.

Table 2: Formatted 2020 Small Launch Provider Data

	Funding ($M)	Launch Cost ($M)	Price ($/kg)
Rocket Lab	257.31	4.9	16333
Astra Space	100.00	2.5	12255
iSpace	275.00	5.0	16667
Firefly Aerospace	23.16	15.0	15000
Relativity Space	684.54	10.0	8000
ABL Space Systems	49.00	12.0	8889
One Space	116.00	3.2	16000

System Description

In this report we will examine the effect that the level of funding for these space launch ventures has on each company's capabilities in terms of cost per launch and cost per kilogram of mass launched. It would also be interesting to analyze how funding input affects the total payload capacity, however this study will be focused on minimizing inputs, whereas the total payload capacity data would be better suited for an analysis on maximizing output (another time, perhaps).

The below diagram depicts the system of interest in terms of the inputs and outputs being considered in this analysis:

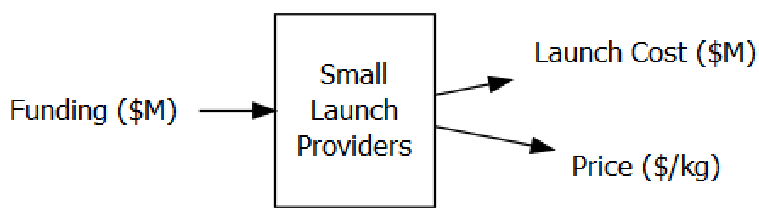

Figure 1: Funding impact on Small Launch Provider capabilities

The questions that we want to answer regarding this system are:

- Which of these companies promises the best performance?
- Which of these companies are laggards?
- For those that are under-performing, are there objective performance targets they should be able to reach?
- Which companies might be employing best practices that could be adopted by other companies?
- Among the companies that are lagging in performance, are there particular other companies they should focus attention on for learning best practices?

Let's build our model and explore these questions.

Developing our Model (Mathematical Formulation)

As mentioned in the text for this course, *Optimization Modeling Using R*, "A key way to begin the mathematical development of the envelopment model is to ask, can you find a combination of units that produces a target with at least as much output using less input?"

We want to look at the level of funding input for these companies to determine if the launch capabilities are optimal after picking the best performing of these companies as a benchmark.

Following with the conventions outlined in chapter 5 of the text, let N^X (or NX in our coded model) be the number of inputs into our model; NY, or NY, will be the number of outputs from the model. Finally, N^D, or ND, will denote the number of DMU's (decision making units). Then we want to construct a vector λ that describes the minimal inputs to achieve the same level of output as the DMU k. Then the value of the ith input used is $\sum_{j=1}^{N^D} x_{i,j}\lambda_j$, while the value of the rth output produced is $\sum_{j=1}^{N^D} y_{r,j}\lambda_j$. Note that here each λ_j is one of the companies in our analysis.

Now we need to set the objective function, which will be determined based on the *input orientation*, meaning that we are trying to achieve efficiency through input reduction. Colloquially, this translates to attempting to achieve the same launch capabilities with less funding (cost-efficiency).

Let θ, which represents the radial reduction in DMU (k's) input, be the value for input usage. A score of $\theta = 1$ denotes that no input reduction is possible, conversely if $\theta = .9$ then there is an opportunity to reduce input by 10%. So our objective is to minimize θ as much as possible. Let the following formula represent our model thus far

$$
\begin{aligned}
&\sum_{j=1}^{N^D} x_{i,j}\lambda_j \leq x_{i,k} \ \forall \ i \\
&\sum_{j=1}^{N^D} y_{r,j}\lambda_j \geq y_{r,k} \ \forall \ r \\
&\lambda_j \geq 0 \ \forall \ j
\end{aligned}
$$

One final concern that we need to consider in developing this model is scalability. Indeed, this is the main reason that we have excluded larger companies such as Blue Origin and SpaceX. However even when we restrict ourselves to small launch providers there is a lot of variability in the scale of the companies and vehicles offered. An important question that arises in this analysis then is if the factors that work for one company's success can even scale work with another company of a different size?

For this reason we will utilize four methodologies to look at the impact that scale has on our results. These are:

(1) First is the constant returns to scale (CRS). This is often referred to in the DEA literature as CCR after Charnes, Cooper and Rhodes. In this paper, we will refer to it as CRS for simplicity and consistency in making reference to *returns to scale*. This is the simplest methodology and requires no changes to what we have already formulated.

(2) Second is the variable returns to scale (VRS). This is often referred to in the DEA literature as BCC after Banker, Charnes, and Cooper, but in this paper we will use VRS.

(3) Next is the increasing returns to scale (IRS, also referred to as "non-decreasing returns to scale"). We will refer to it as IRS.

(4) Finally is the decreasing returns to scale (DRS, also referred to as "non-increasing returns to scale").

In the table below we see the formulation for each version of the returns to scale restraint:

Table 3: Returns to Scale Constraint

Returns to Scale	Envelopment Constraint
CRS	No constraint needed
VRS	$\sum_{j=1}^{N^D} \lambda_j = 1$
IRS/NDRS	$\sum_{j=1}^{N^D} \lambda_j \geq 1$
DRS/NIRS	$\sum_{j=1}^{N^D} \lambda_j \leq 1$

These return to scale variants will be examined one at a time once we enter our model into OMPR (the optimization engine that we're using for this analysis) using the Returns to Scale variable, denoted as **RTS**.

Now that we've specified our model elements, let's take a look at the formulation. Note that we will operate with the following constraints which must all be satisfied simultaneously. Also

Then we can formulate our general linear programming model as:

$$\begin{aligned} \text{minimize } & \theta \\ \text{subject to } & \sum_{j=1}^{n} \lambda_j = 1 \\ & \sum_{j=1}^{n} x_{i,j}\lambda_j - \theta x_{i,k} \leq 0 \; \forall \; i \\ & \sum_{j=1}^{n} y_{r,j}\lambda_j \geq y_{r,k} \; \forall \; r \\ & \lambda_j \geq 0 \; \forall j \end{aligned}$$

Note that we'll substitute the appropriate RTS constraint as appropriate to our analysis in OMPR.

Building our Model in OMPR

Now that we've determined the general structure of our model, we'll build it in OMPR so that we can iterate over the number of DMUs under consideration (this allows us to add new companies to the database without needing to recode for it). We'll also set it up in a modular fashion so that the same code can be copy/pasted and all we need to specify is which version of the RTS constraint we are using.

The Model with RTS = CRS

```
RTS<-"CRS"
for (k in 1:ND) {

  result <- MIPModel() %>%
    add_variable(vlambda[j], j = 1:ND, type="continuous",
                 lb = 0) %>%
    add_variable(vtheta, type = "continuous") %>%
    set_objective(vtheta, "min") %>%
    add_constraint(sum_expr(vlambda[j] * xdata[j,i],
                            j = 1:ND)
                   <= vtheta * xdata[k,i], i = 1:NX,
                   .show_progress_bar=FALSE ) %>%
    add_constraint(sum_expr(vlambda[j] * ydata[j,r],
                            j = 1:ND,
                   .show_progress_bar=FALSE )
                   >= ydata[k,r], r = 1:NY)
    if (RTS=="VRS") {result <-
      add_constraint(result, sum_expr(vlambda[j],
                                      j = 1:ND) == 1) }
    if (RTS=="IRS") {result <-
      add_constraint(result, sum_expr(vlambda[j],
                                      j = 1:ND) >= 1) }
    if (RTS=="DRS") {result <-
      add_constraint(result, sum_expr(vlambda[j],
                                      j = 1:ND) <= 1) }
  res <- solve_model(result, with_ROI(solver = "glpk"))

  print(c("DMU=",k,solver_status(res)))

    res.efficiency[k] <- get_solution(res, vtheta)
    res.lambda[k,] <- t(as.matrix(as.numeric(
                     get_solution(res, vlambda[j])[,3] )))
}
```

```
## [1] "DMU="    "1"       "optimal"
## [1] "DMU="    "2"       "optimal"
## [1] "DMU="    "3"       "optimal"
## [1] "DMU="    "4"       "optimal"
## [1] "DMU="    "5"       "optimal"
## [1] "DMU="    "6"       "optimal"
## [1] "DMU="    "7"       "optimal"
```

```
SLP_CRS.Res <- cbind(res.efficiency, poscol(res.lambda))
colnames(SLP_CRS.Res)[1] <- paste(RTS, '- IO')
colnames(SLP_CRS.Res)[2] <- launch2020[4,2]
```

Discussion of Results

First we confirm that the solver produces optimal results for each DMU, as seen in the printout above. Then as per the table, note that I've used the poscol() helper function from the TRA library to eliminate all zero columns, leaving us with only the results that returned efficiency scores.

Table 4: Results from Small Launch Providers with Constant Rate of Returns (CRS)

	CRS - IO	Firefly Aerospace
Rocket Lab	0.098	1.089
Astra Space	0.189	0.817
iSpace	0.094	1.111
Firefly Aerospace	1.000	1.000
Relativity Space	0.023	0.667
ABL Space Systems	0.378	0.800
One Space	0.213	1.067

In this case, the first column shows us how each company compares to the highest performing company(s) as determined by the model and the second column shows the how the benchmark compares to each company. We see that OMPR has selected *Firefly Aerospace* (Firefly) as the singular benchmark. This means that using itself as a benchmark, Firefly scores $\lambda = 1$ and no input reduction is possible (according to this model) to reach it's current level of output.

With that established, let's look at the scores for the other companies. Looking at Astra Space's score in the first column, we read that if Firefly was rescaled to the same level of output as Astra, it would need $100 - 18.9 = 81.1\%$ less input. We can also state this such that Astra Space performs at 18.9% efficiency compared to the benchmark. Conversely, the model states that the output performance of iSpace would be surpassed by a version of Firefly that is 11.1% larger than it currently is.

These results are interesting but suspect. We may consider that since space launch operations and vehicles are very expensive, scalability is an important factor probably shouldn't be assumed to be constant. Further, if I graph the efficiency frontier based on constant returns to scale (below), we see no upper bound that is reachable by the competing companies. Given this information, let's run the model in OMPR with RTS set to **VRS** for variable returns to scale.

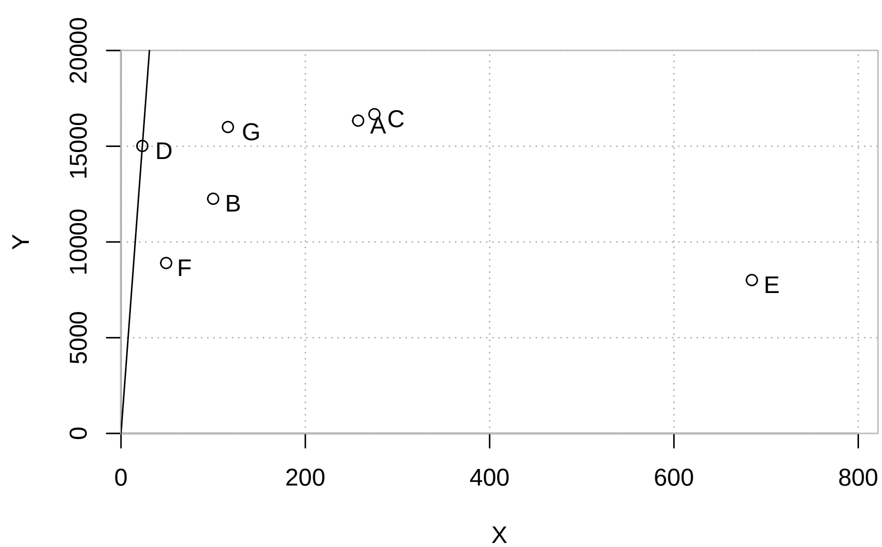

Figure 2: Frontier Plot with Constant Returns to Scale

The Model with RTS = VRS

Note: since the modularized code is the same except for changing the RTS variable, I will not show the code chunks for the remaining analyses. However I will continue to display all of the outputs so we can be sure that the analysis was performed correctly.

```
## [1] "DMU="    "1"       "optimal"
## [1] "DMU="    "2"       "optimal"
## [1] "DMU="    "3"       "optimal"
## [1] "DMU="    "4"       "optimal"
## [1] "DMU="    "5"       "optimal"
## [1] "DMU="    "6"       "optimal"
## [1] "DMU="    "7"       "optimal"
```

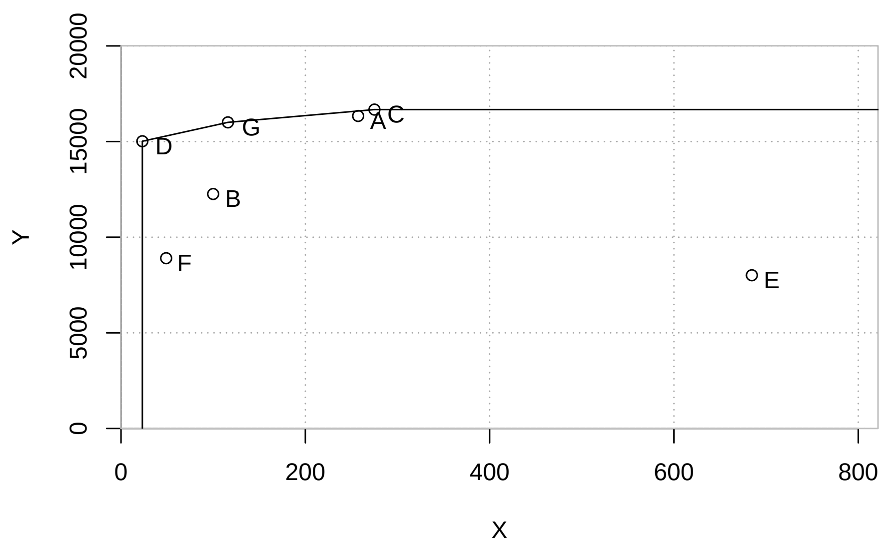

Figure 3: Frontier Plot with Variable Returns to Scale

Table 5: Results from Small Launch Providers with Variable Returns to Scale (VRS)

	VRS - IO	iSpace	Firefly Aerospace	One Space
Rocket Lab	0.791	0.582	0.055	0.363
Astra Space	0.232	0.000	1.000	0.000
iSpace	1.000	1.000	0.000	0.000
Firefly Aerospace	1.000	0.000	1.000	0.000
Relativity Space	0.034	0.000	1.000	0.000
ABL Space Systems	0.473	0.000	1.000	0.000
One Space	1.000	0.000	0.000	1.000

Discussion of Results

With the Variable Returns to Scale constraint, we see in the graph that the **Frontier Plot** is much more agreeable, and seemingly attainable. Interestingly, it appears that while Firefly is again a top performer, other companies are now given efficiency scores and we even see three more companies (*Rocket Lab*, the only of these companies to have commercial operations; and the two Chinese companies *iSpace* and *One Space*) at or near the efficiency frontier. This is actually not too surprising since we should expect a company that has been running operations successfully for multiple years (Rocket Lab) to be approaching it's own version of efficiency, and of the remaining companies, Firefly is the nearest to beginning it's own operations later this year (2021).

Also interesting is that according to the plot there is one significant outlier: *Relativity Space*. If we refer back to the original data table, we see that this company has received nearly 2.5 times more funding than the next highest funded company ($684.5 M vs $275M). Relativity Space also promises the lowest cost per kilogram yet has toward the high end of overall launch cost. Given these factors, we don't even need to run the OMPR solver to see that this is the least efficient company by far.

Now let's look at the solver output table. We see that Firefly, iSpace and One Space, which were determined to be right on the efficiency frontier in the graph, are now the three benchmarks chosen by the solver in the table. We can interpret these results in the same way as with the CSR model. Rocket Lab scores at 79.1% efficiency, while according to these benchmarks Relativity Space scores only at 3.4%. Other companies rate somewhere in-between.

The Model with RTS = DRS

```
## [1] "DMU="    "1"       "optimal"
## [1] "DMU="    "2"       "optimal"
## [1] "DMU="    "3"       "optimal"
## [1] "DMU="    "4"       "optimal"
## [1] "DMU="    "5"       "optimal"
## [1] "DMU="    "6"       "optimal"
## [1] "DMU="    "7"       "optimal"
```

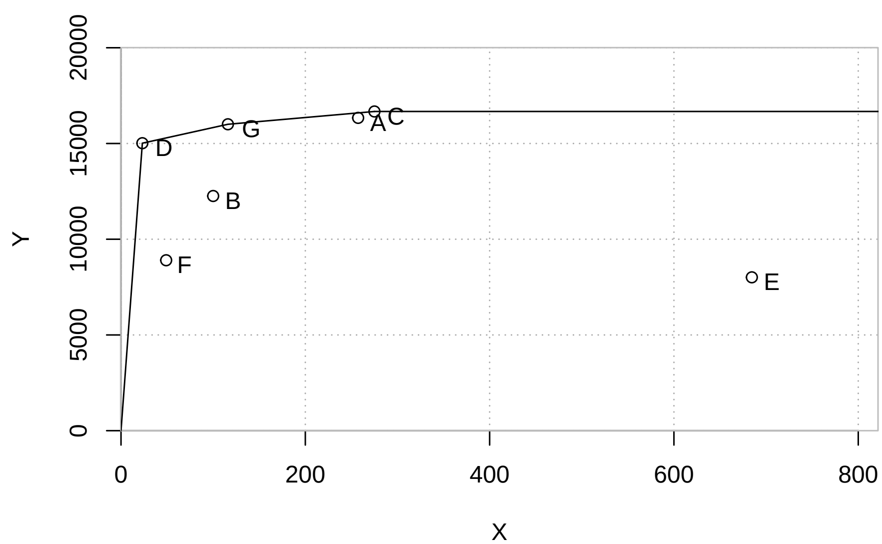

Figure 4: Frontier Plot with Decreasing Returns to Scale

Table 6: Results from Small Launch Providers with Decreasing Returns to Scale (DRS)

	DRS - IO	iSpace	Firefly Aerospace	One Space
Rocket Lab	0.791	0.582	0.055	0.363
Astra Space	0.189	0.000	0.817	0.000
iSpace	1.000	1.000	0.000	0.000
Firefly Aerospace	1.000	0.000	1.000	0.000
Relativity Space	0.023	0.000	0.667	0.000
ABL Space Systems	0.378	0.000	0.800	0.000
One Space	1.000	0.000	0.000	1.000

Discussion of Results

The analysis returns very similar results when I set RTS to Decreasing Returns to Scale, but we do seem to get a little more information. The efficiency frontier is the same as with the VRS model but we get slightly different values particularly when comparing Firefly to other companies, whereas with the VRS model it was rated 1/1 with Astra, Relativity and ABL, now those values are more interesting. Given the shape of shape of the efficiency frontier, this seems to be the most appropriate model. However since the most interesting information it provides is similar to what we saw in VRS, we will move on for now.

The Model with RTS = IRS

```
## [1] "DMU="    "1"       "optimal"
## [1] "DMU="    "2"       "optimal"
## [1] "DMU="    "3"       "optimal"
## [1] "DMU="    "4"       "optimal"
## [1] "DMU="    "5"       "optimal"
## [1] "DMU="    "6"       "optimal"
## [1] "DMU="    "7"       "optimal"
```

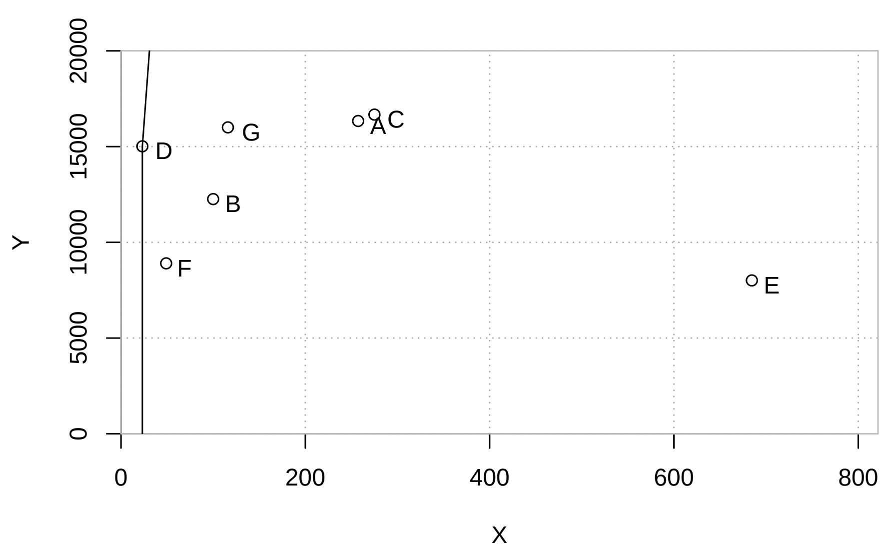

Figure 5: Frontier Plot with Increasing Returns to Scale

Table 7: Results from Small Launch Providers with Increasing Returns to Scale (IRS)

	IRS - IO	Firefly Aerospace
Rocket Lab	0.098	1.089
Astra Space	0.232	1.000
iSpace	0.094	1.111
Firefly Aerospace	1.000	1.000
Relativity Space	0.034	1.000
ABL Space Systems	0.473	1.000
One Space	0.213	1.067

Discussion of Results

In this final model, we see a similar efficiency frontier mapping as with the CRS model and very similar results. Here we will suffice it to say that the IRS analysis seems to be about as appropriate for understanding this dataset as the CRS analysis.

Comparison of Aggregated Results

Table 8: Comparison of Efficiency Scores

	CRS - IO	VRS - IO	DRS - IO	IRS - IO
Rocket Lab	0.098007	0.790583	0.790583	0.098007
Astra Space	0.189217	0.231600	0.189217	0.231600
iSpace	0.093578	1.000000	1.000000	0.093578
Firefly Aerospace	1.000000	1.000000	1.000000	1.000000
Relativity Space	0.022555	0.033833	0.022555	0.033833
ABL Space Systems	0.378122	0.472653	0.378122	0.472653
One Space	0.212966	1.000000	1.000000	0.212966

Final Discussion

In comparing the results of all four analyses side by side, we can see the similarities and differences in the different approaches. With the given data available, Firefly rated as the most efficient small launch provider in every analysis performed in this study, however we determined that the most appropriate analysis seems to have come from the Decreasing Returns to Scale model, with similar results from the Variable Returns to Scale model. Part of how we made this determination is that we know that Rocket Lab is a competitive launch provider, and thus must be operating at a decent level of efficiency. A nearly 80% efficiency rating seems more likely with what we know that a 10% efficiency rating.

Interestingly, regardless of which model we choose we see that Relativity Space, which has received the highest level of funding, is by far the least efficient. This doesn't take into account that this particular company is developing novel technology (near-completely 3D printed rockets) but it is accounting for both the cost per launch and the cost per kilogram. Most of the other companies have a much higher cost per kilogram, but a much lower cost per launch. This would seem to suggest that Relativity Space intends to undercharge its customers as it enters the market. However when we look at the payload capacity, we see that it has the second highest of the companies examined, and that may help account for this seeming discrepancy. As with the other companies besides Rocket Lab, we won't be able to say anything conclusively about any of these companies' efficiency ratings until (or if) they have sustained commercially viable operations.

This may be grounds for a future study as more data in this new market becomes available.