Supplementary Material for "A Multi-Objective Capacity-Constrained Optimization of Corn Planting Scheduling"

1. Long Short-Term Memory Model Details

There are special "cells" in LSTM, called LSTM units, in each hidden layer of the neural network, and these cells are connected sequentially. LSTM apply both hidden state and cell state in its interval architecture, where hidden state represents the current "memory" of the network based on the input and the information of the previous cell, and cell state is the memory component of LSTM for storing information over long-time period. As comparison, traditional RNN only utilizes hidden state in its architecture. The modification of LSTM over traditional RNN makes it able to establish a mechanism so that each cell has an input gate, a forget gate, and an output gate, and these gates will together determine the utilization of information from the previous cell, the memorization of information in current cell, the output of the current cell, and the information that should be passed to the next cell. This sequence of output is either passed into the next hidden layer for further processing, or processed by a regular deep learning output layer (called the Dense layer) to produce a final prediction. Compared to the traditional RNN, LSTM models remember a larger number of steps in the sequence via the memory gates (Goodfellow et al., 2016).

There are several hyperparameters that need to be specified for the LSTM model, including the number of epochs, batch size and number of hidden layers. An epoch is a complete pass through the entire training dataset, the batch size is the number of training samples randomly drawn from the training dataset and fed into the neural network model at one time (Siami-Namini et al., 2018), and the number of hidden layers is the number of layers in the network which neither directly receive data input nor output the final result Goodfellow et al. (2016). As a neural network model, LSTM uses gradient descent to optimize its internal parameters across epochs. Based on informal testing, we chose the batch size to be 5 with 80 epochs. Tables **??** and **??** suggest that, compared with 3 hidden layers, 2 hidden layers result in a model that predicts well. Thus, the process of training the model would be as follows. Randomly select 5 sequences, and use them to train the model. Repeat this process until all of the sequences in the training set are exhausted. This constitutes one epoch. Repeat this process for a total of 80 epochs. Our LSTM framework, then, is composed of one Input Layer, followed by two LSTM hidden layers, in which the first LSTM hidden layer has K_1 LSTM units ("cells" in our informal description above), the second LSTM hidden layer has K_2 LSTM units, and a Dense layer to produce predicted GDU from the output of the second LSTM hidden layer. As shown in Figure 1, the Input Layer takes the sequence $g_{y-l,md}, \ldots, g_{y-2,md}, g_{y-1,md}$ as input and feeds the input into the first LSTM hidden layer. Using the notation of James et al. (2021), the first LSTM hidden layer takes the input to generate the intermediate output of the first LSTM hidden layer, $A^{(1)}$ which is a two-dimensional matrix composed of l rows and K_1 columns. The second LSTM hidden layer uses the two-dimensional matrix $A^{(1)}$ as input to generate a vector $A^{(2)}$ which has K_2 elements. The Dense Layer uses the vector $A^{(2)}$ to compute the predicted $\hat{g}_{ymd} = \beta_0 + \sum_{k=1}^{K_2} \beta_k A_k^{(2)}, k = 1, \ldots, K_2$. Based in part on informal testing, for our implementation of the model with 2 hidden layers, we chose $K_1 = 64$ and $K_2 = 16$. (For the model implementation with 3 hidden layers, we used $K_1 = 200$, $K_2 = 64$ and $K_3 = 16$.)

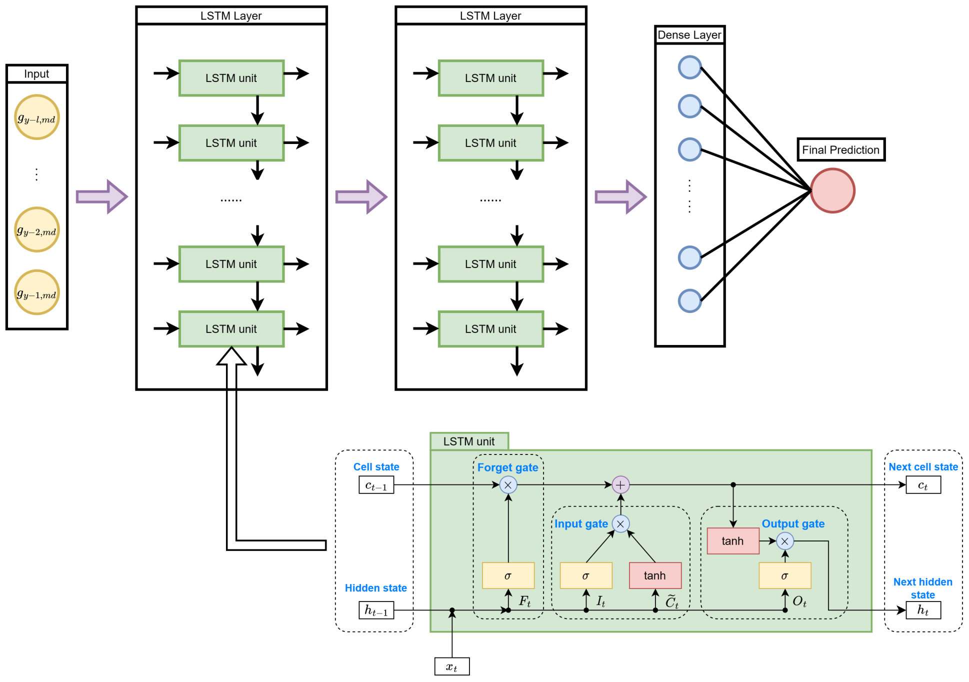


Figure 1: Compared to traditional RNN, LSTM units are composed of hidden state, cell state and different types of gates: input gates, forget gates and output gates. These states and gates cooperatively determine the flow of information within the sequentially connected LSTM units (Altché & de La Fortelle, 2017, Figure 4). The input layer takes the GDU sequence as input and passes the input into the first LSTM hidden layer. Then the first LSTM hidden layer produces a two-dimensional matrix as intermediate output and the second LSTM hidden layer uses the two-dimensional matrix to produce an intermediate vector. Finally, the Dense layer uses the intermediate vector to produce the final GDU prediction. More details regarding the LSTM units can be obtained in Altché & de La Fortelle (2017) and Gers et al. (1999)

2. Details for Scenario 1, Site 0 Results

Here we provide details, cross-referenced with the steps outlined in Section 3.2.1, to demonstrate the results for Scenario 1, Site 0. As in Step 3, PF_1 has 4,144 solutions in the constructed Pareto front, while PF_2 has 821 and PF_3 has 1,152. Combining them together results in a meta-population, P_{PF}, which yields a 6117×4 matrix. In order to compute the hypervolume (Step 4), we found that $F_1^{min} = (6, 1027, 49, 12788)$ and $F_1^{max} = (1316, 8957, 54, 48764)$, which are used to scale each solution in PF_k, $k = 1, 2, 3$. As in Step 5, the hypervolume v_k is computed for each of the scaled Pareto fronts, PF_k, such that $v_1 = 11.5$, $v_2 = 15.6$, and $v_3 = 13.4$. This suggests that when evaluated on the Model 1 objectives, the Model 2 Pareto front produces

Table 1: Criterion values for solutions $\mathbf{p}^*$ (final single optimal solution), $\mathbf{p}_{challenge}$ (solution submitted to the Challenge), $\mathbf{p}_{initial}$ (initial solution given by the Challenge), under Scenario 1, site 1

	$\mathbf{p}^*$	$\mathbf{p}_{challenge}$	$\mathbf{p}_{initial}$
Median Absolute Difference ($\mathbf{f}_{11}$)	8	365	2,196
Max Absolute Difference ($\mathbf{f}_{12}$)	827	1,816	6,725
# of Non-zero Harvest Week ($\mathbf{f}_{13}$)	51	53	52
Total Amount of Wasted Product ($\mathbf{f}_{14}$)	4,809	12,290	59,549

the best set of solutions, as measured by the hypervolume. Thus, we will choose our final solution from PF_2, using TOPSIS (Step 6). That is, after the scaling and the weighting, we compute PF_2's positive ideal solution as $A^+ = (0.0002, 0.0023, 0.0085, 0.0053)$ and its negative ideal solution $A^- = (0.0259, 0.0182, 0.0090, 0.0142)$.

3. Results for Scenario 1 (Site 1)

We first provide the real and predicted GDU values for Site 1 (Figure 2), analogous to Figure 1 in the main document.

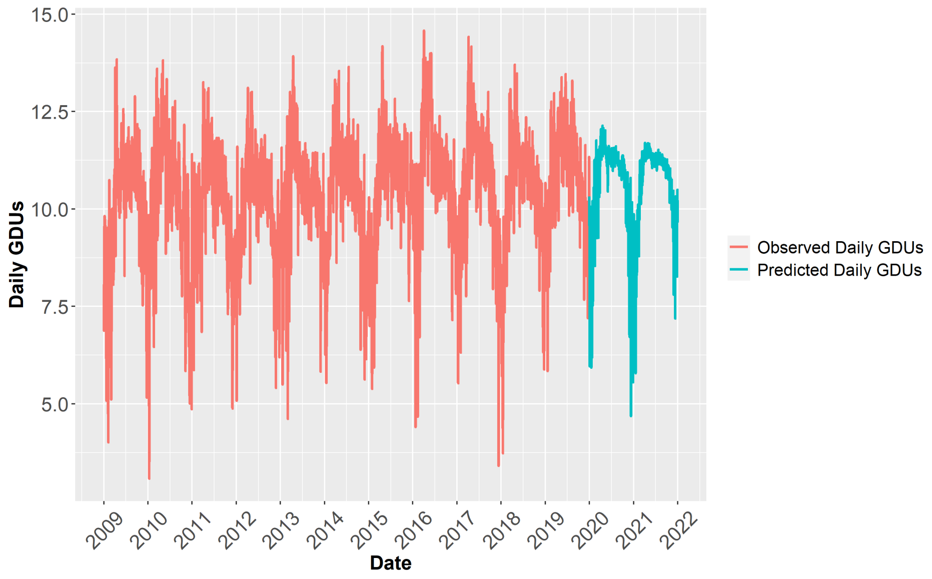
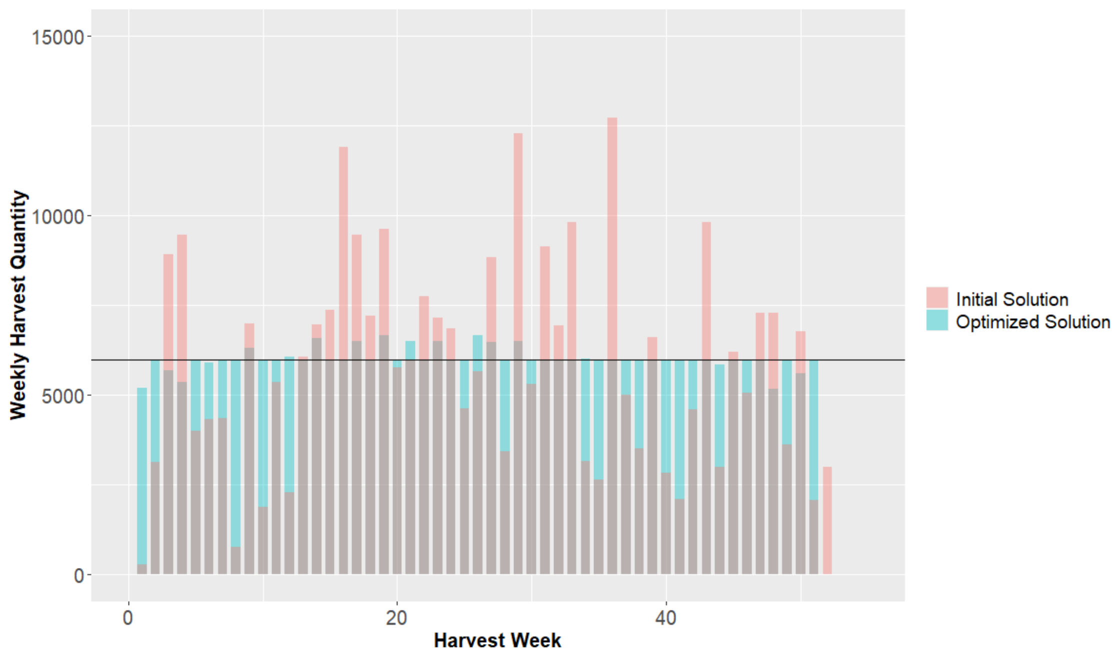
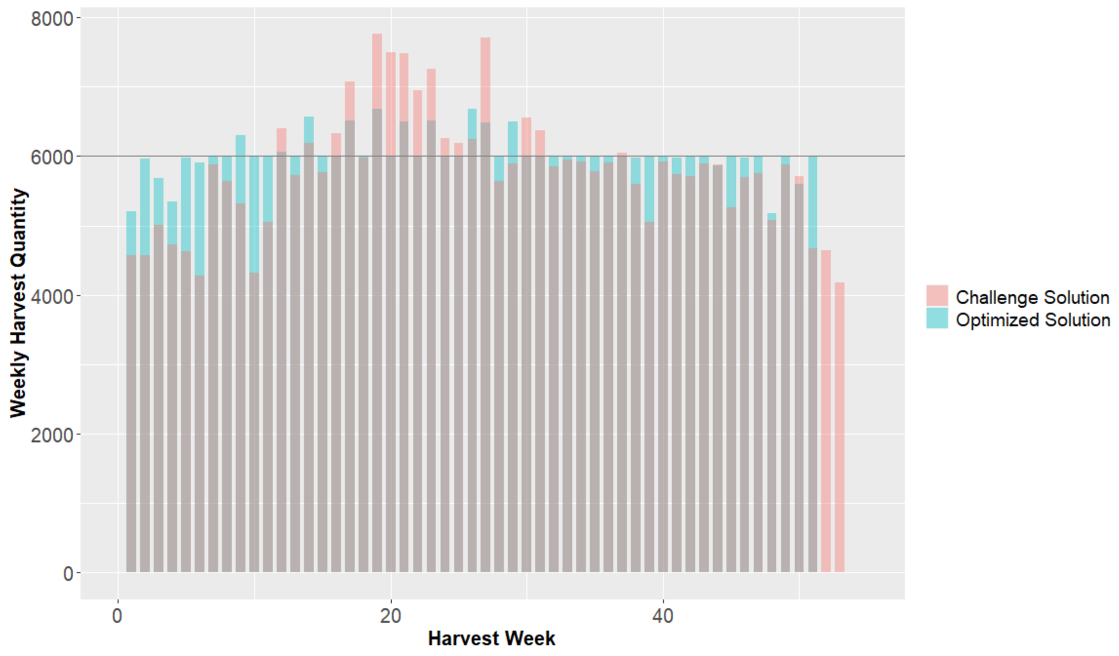


Figure 2: For Site 1, real GDU values between 2009 and 2019, and predicted GDU values for 2020 and 2021, using the LSTM(2,80) model.

Omitting most details, we focus here on the results themselves. For Site 1, Scenario 1, there is a designated storage capacity of 6,000 ears per week. Using the methodology outlined in Section 4.2 of the main document, we find again that the Pareto front constructed using Model 2 yields the best hypervolume, leading to a final solution given in Table 1 and Figure 3.



(a) Comparison between optimized and original planting schedule, where the original was provided as an initial solution for the Challenge.



(b) Comparison between the solution submitted to the Challenge and the solution optimized using the methods described in this paper.

Figure 3: Scenario 1, Site 1

4. Scenario 2 Methodology

For Scenario 2 we have decision variables p_i, $i = 1, 2, \ldots, n_0$, representing the planting day for seed population s_i, as well as $\hat{C}$, a decision variable representing the location capacity. As in Section 3.2, the problem requires $L(s_i) \leq p_i \leq U(s_i)$, but we introduce a new set of constraints having to do with the capacity. Specifically, we require $L(C) \leq \hat{C} \leq U(C)$, where $L(C)$ and $U(C)$ are a lower bound and upper bound that is user-specified based upon the particular setting. In addition to adding $\hat{C}$ as a decision variable, we also include it as its own objective function, in order that we might select a capacity as small as possible, while simultaneously balancing the number of harvest weeks, the deviation from capacity, and the amount of wasted corn. More specifically, for Models 1, 2, and 3 in Section 3.2, we augment the existing objective functions with a new one: $\mathbf{f}_{k5}(\mathbf{p}) = \hat{C}$, for $k = 1, 2, 3$. Hence, we are trying to simultaneously minimize $\mathbf{f}_k(\mathbf{p}) = (\mathbf{f}_{k1}(\mathbf{p}), \mathbf{f}_{k2}(\mathbf{p}), \mathbf{f}_{k3}(\mathbf{p}), \mathbf{f}_{k4}(\mathbf{p}), \mathbf{f}_{k5}(\mathbf{p}))$, $k = 1, 2, 3$, with decision variables $\mathbf{p}$ and $\hat{C}$.

Beyond these generalizations of the decision variables and objective functions, our methodology remains largely unchanged. The Pareto dominance description (Section 4.2.1) is the same, except it now accounts for the extra decision variable and objective function. The hypervolume computation simply incorporates the new objective and extends the reference point to $[2, 2, 2, 2, 2]$. As before, though we have defined $\mathbf{f}_2(\mathbf{p})$ (Model 2) and $\mathbf{f}_3(\mathbf{p})$ (Model 3), they only serve to aid in the construction of potentially improved Pareto fronts with respect to $\mathbf{f}_1(\mathbf{p})$ (Model 1). Our strategy to explore the tuning parameter space for Scenario 2 is also the same as for Scenario 1. Based on this structure, we use the same six-step solution strategy as outlined in Section 4.2.1, except that we account for the fifth objective and the additional decision variable.

We note that due to the curse of dimensionality, adding a fifth criterion makes the computation much more difficult. We also note that for the selection of the final single solution $\mathbf{p}^*$ from the best Pareto front PF_{k*}, using TOPSIS, instead of using equal weights for all five objective functions, we use weights $\{w_1 = 0.225, w_2 = 0.225, w_3 = 0.225, w_4 = 0.225, w_5 = 0.1\}$. The slightly lower weight for the new objective reflects our belief that the other objectives should be prioritized. Once $\mathbf{p}^*$ is chosen, the corresponding $\mathbf{f}_{15}(\mathbf{p}^*) = \hat{C}$ is our recommended location capacity.

4.1. Results for Scenario 2 (Site 0)

As described in Section 4.3, Scenario 2 is like Scenario 1 except a capacity is not specified. Instead, as part of the solution, we need to provide a recommended capacity. Here we provide results for site 0; results for site 1 could be obtained similarly, but are omitted because the hypervolume computation cost for the resulting large Pareto front is high.

Based on Figure 1 we observe that, on average, the population harvest quantity under Scenario 2 is larger than Scenario 1 which suggests that we need a relatively larger location capacity under Scenario 2. Thus, we specify $L(C) = 8{,}000$ and $U(C) = 20{,}000$, where the latter is chosen to be larger than we anticipate being necessary. Using the methodology described in Section 4.3, Table 2 shows our chosen $\mathbf{p}^*$. Since the initial Syngenta solution for Scenario 2 did not have a corresponding estimated location capacity, here we only

Table 2: Criterion values for solutions $\mathbf{p}^*$ (final single optimal solution) and $\mathbf{p}_{challenge}$ (solution submitted to the Challenge), under Scenario 2, site 0

	$\mathbf{p}^*$	$\mathbf{p}_{challenge}$
Median Absolute Difference ($\mathbf{f}_{11}$)	409.599	1,836
Max Absolute Difference ($\mathbf{f}_{12}$)	4,286.599	3,402
# of Non-zero Harvest Week ($\mathbf{f}_{13}$)	51	53
Total Amount of Wasted Product ($\mathbf{f}_{14}$)	11,639.210	37,421
Estimated Location Capacity ($\mathbf{f}_{15}$)	10,312.6	9,864

compare the single optimal solution $\mathbf{p}^*$ with the solution we originally submitted to the Syngenta challenge. Compared to $\mathbf{p}_{challenge}$, we suggest a capacity of about 450 more ears of corn. This slight concession results in a much smaller median absolute difference and amount of wasted product, though our proposed solution has a larger maximum absolute difference. Our solution $\mathbf{p}^*$ also uses two fewer harvest weeks. The solutions can be compared in Figure 4.

References

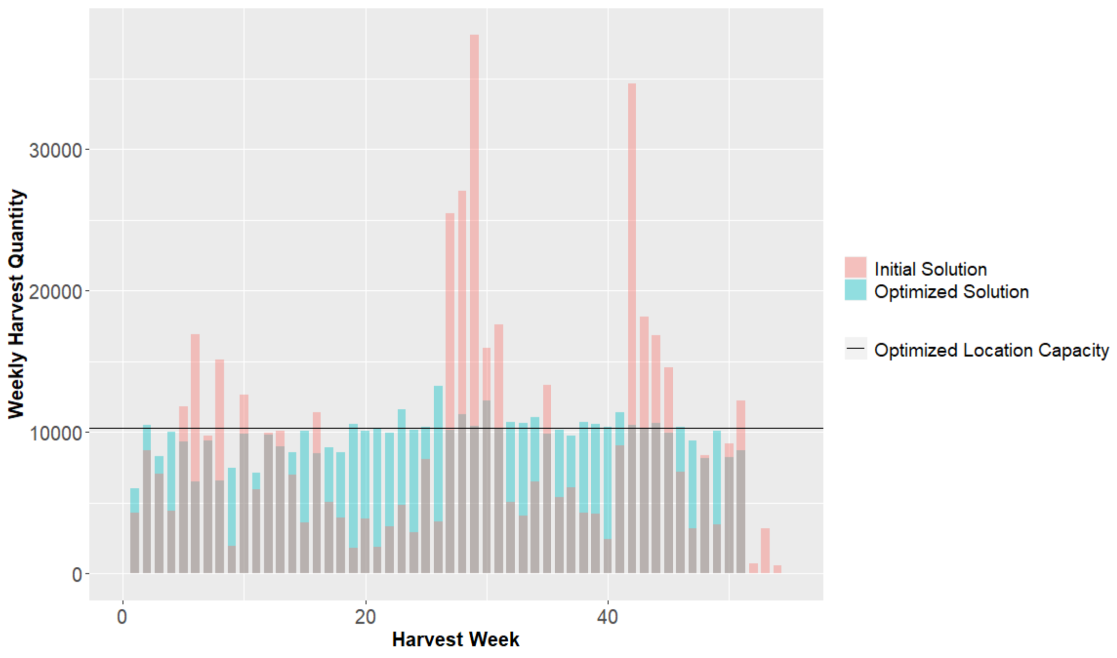
Altché, F., & de La Fortelle, A. (2017). An lstm network for highway trajectory prediction, . (p. 353–359). URL: `https://doi.org/10.1109/ITSC.2017.8317913`. doi:`10.1109/ITSC.2017.8317913`.

Gers, F., Schmidhuber, J., & Cummins, F. (1999). Learning to forget: continual prediction with lstm. In *1999 Ninth International Conference on Artificial Neural Networks ICANN 99. (Conf. Publ. No. 470)* (pp. 850–855 vol.2). volume 2. doi:`10.1049/cp:19991218`.

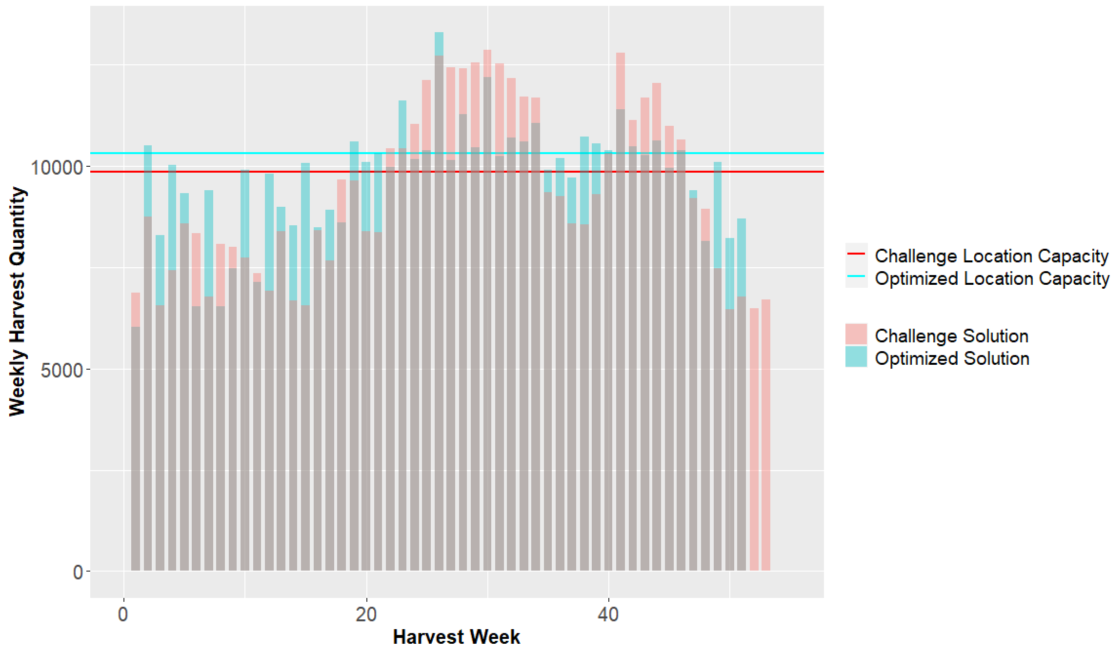
Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press. `http://www.deeplearningbook.org`.

James, G., Hastie, T. J., Tibshirani, R., Witten, D., & James, G. (2021). *An introduction to statistical learning: With applications in R*. Springer.

Siami-Namini, S., Tavakoli, N., & Siami Namin, A. (2018). A comparison of arima and lstm in forecasting time series. In *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)* (pp. 1394–1401). doi:`10.1109/ICMLA.2018.00227`.



(a) Comparison between optimized and original planting schedule, where the original was provided as an initial solution for the Challenge.



(b) Comparison between the solution submitted to the Challenge and the solution optimized using the methods described in this paper.

Figure 4: Scenario 2, Site 0