Towards Intelligent Food Waste Prevention: An Approach Using Scalable and Flexible Harvest Schedule Optimization with Evolutionary Algorithms

MAURICE GÜNDER1,2, NICO PIATKOWSKI1, LAURA VON RUEDEN2, RAFET SIFA2, AND CHRISTIAN BAUCKHAGE1,2

1 Institute for Computer Science III, University of Bonn, Friedrich-Hirzebruch-Allee 5, 53115 Bonn, Germany
2 Fraunhofer Institute for Intelligent Analysis and Information Systems IAIS, Schloss Birlinghoven, 53757 Sankt Augustin, Germany

Corresponding author: Maurice Günder (e-mail: mguender@uni-bonn.de).

This work was partially funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany’s Excellence Strategy – EXC 2070 – 390732324. Other parts were funded by the Federal Ministry of Education and Research of Germany as part of the competence center for machine learning ML2R (01IS18038B).

ABSTRACT In times of climate change, growing world population, and the resulting scarcity of resources, efficient and economical usage of agricultural land is increasingly important and challenging at the same time. To avoid disadvantages of monocropping for soil and environment, it is advisable to practice intercropping of various plant species whenever possible. However, intercropping is challenging as it requires a balanced planting schedule due to individual cultivation time frames. Maintaining a continuous harvest throughout the season is important as it reduces logistical costs and related greenhouse gas emissions, and can also help to reduce food waste. Motivated by the prevention of food waste, this work proposes a flexible optimization method for a full harvest season of large crop ensembles that complies with given economical and environmental constraints. Our approach applies evolutionary algorithms and we further combine our evolution strategy with a sophisticated hierarchical loss function and adaptive mutation rate. We thus transfer the multi-objective into a pseudo-single-objective optimization problem, for which we obtain faster and better solutions than those of conventional approaches.

INDEX TERMS adaptive mutation, evolution strategy, evolutionary algorithm, Gaussian process regression, harvest schedule, hierarchical loss, optimization

I. INTRODUCTION

In a recent study, the Food and Agriculture Organization of the United Nations stated that almost 1.4 billion hectares or 30% of the world’s agricultural land are used to produce food that later goes to waste. They also stated that global food loss causes 3.5 gigatons of carbon dioxide emissions or 7% of the world’s total greenhouse gas emissions [FAO, 2013], [FAO, 2019]. Efficient use of agricultural and logistical capacities are therefore crucial for sustainable global food supply.

Developments in remote sensing and field robotics enable the use of computational intelligence in agriculture [Bauckhage and Kersting, 2013], [Kamilaris and Prenafeta-Boldú, 2018]. In combination with advances in machine learning, computer vision, and geographic information systems, these are key drivers for the branches of precision agriculture and precision farming. The applications are manifold, e.g. via plant phenotyping [Yandun Narvaez et al., 2017] or livestock farming [Monteiro et al., 2021].

Moreover, contemporary precision agriculture applies machine learning for the optimization of cost-benefit-based objectives to maximize yield [Lin et al., 2018] or to ensure resilience against environmental or economical stresses [Singh et al., 2016], [Storm et al., 2019]. A particular application of artificial intelligence in agriculture is the optimization of harvest schedules or agri-food supply chains [Taşkın et al., 2018].
Bilgen, 2021]. Even in other food-related disciplines, harvest schedules are of major importance, e.g. in aquaculture [Yu and Leung, 2006].

In this work, we will focus on optimizing a planting schedule with regard to a sustainable yield sketched in Figure 1. In particular, our goal is to find planting dates for many crop species such that the resulting yield is continuously distributed. In this way, we facilitate efficient prevention of food waste and surplus greenhouse gas emissions.

These kinds of scheduling problems are difficult to solve by involving numerous soft- and hard constraints and requiring to factor in environmental conditions. Common approaches thus consider stochastic models and commercial solvers which, however, are essentially “black box” systems whose decisions are opaque and difficult to comprehend for practitioners in the fields. Exact solvers can tackle scheduling problems that can be formulated as integer linear programs, so called ILP problems, e.g. in [Santos et al., 2019]. They are commonly cast as multi- or many-objective optimization [Branke et al., 2008]. However, exact methods have the disadvantage that the objective functions are restricted to rather special characteristics that do not satisfy complex and multifaceted problem formulations with sophisticated and flexible objectives. Thus, besides many other optimization strategies like in [Ferrer et al., 2008] and [Varas et al., 2020], scheduling problems are often optimized by evolutionary algorithms (EAs) like NSGA-II [Deb et al., 2002] or MOEA/D [Zhang and Li, 2007] as in [Roghanian and Cheraghalipour, 2019], or in combination with other models [Cheraghalipour et al., 2019].

Evolutionary optimization has already been applied to various engineering related problems in a large range of disciplines [Slowik and Kwasnicka, 2020] and, particularly, in the field of agriculture and agronomy. Examples include crop planning [Adeyemo et al., 2010], land-use management [Datta et al., 2007], or vehicle routing [Mohammed et al., 2017]. They can also be tackled via other search heuristics such as, say, simulated annealing (SA) [Kirkpatrick et al., 1983], [ Dong et al., 2018]. All these approaches exploit that SA and EAs allow for solving multivariate and multi-objective problems with flexible objective formulation even for problems where exact methods fail. Just as any other meta-heuristic, SA and EAs can be expected to find a reasonable approximation of the optimum if sufficient computation time is given. However, solution qualities are limited by computational effort or computing resource availability. Since algorithms based on SA cannot be parallelized, EAs often outperform SA for complex problems [Manikas and Cain, 1996]. In the work at hand, we consider a basic evolution strategy (ES) often referred to as $(1+1)$-ES. It is simple to implement and can be generalized towards $(\mu+\lambda)$-ES which allow for distributed computation.

Our contributions can be summarized as follows:

- we propose a generic formalization of the harvest schedule optimization problem which is agnostic with respect to time scales, crop types, and desired yield
- a novel hierarchical loss function is devised that converts naive multi-objective problems into a single-objective problem, enabling faster optimization
- uniform escape from local optima is facilitated via a new dynamic oscillating mutation rate
- we provide an experimental evaluation on real-world data and make our algorithm and benchmark suite available to the community

We show that our approach tackles food waste by still providing a comprehensible problem formulation and implementation.

## II. NOTATION AND BACKGROUND

The following section will introduce the main concepts as prerequisites for our further work.

### A. GAUSSIAN PROCESS REGRESSION

The first building block of our method is Gaussian process regression (GPR) for time series prediction [Rasmussen and Williams, 2006]. It allows us to include available background knowledge about agricultural time series, e.g. the existence of annual periodicity, other seasonal effects, and trends by choosing appropriate kernel functions. GPR is a kernel-based non-parametric method to infer and extrapolate data points. The Gaussian Process defines a distribution over an infinite space of functions, and the kernel controls the prior probability of functions in this space. In order to obtain the posterior probability, data points from the training or the test set are used, respectively. Since we have a finite number of training points, we work in the finite space of functions.

### B. EVOLUTIONARY ALGORITHMS

Evolutionary algorithms are a class of meta-heuristic optimization algorithms inspired by biological mechanisms [Holland, 1992], [Bäck, 1996]. They are frequently applied to
address NP-hard combinatorial optimization problems. The common procedure is to generate one or more possible solutions in the form of state vectors, called populations. During a generation, the populations, or parents, are mutated, resulting in children. An environment thereafter has to select the best population(s) which themselves are initial states for further mutations in the upcoming generation [Hassanat et al., 2019]. The computational procedure within a generation cycle is referred to as evolution strategy. The type of ES we will focus on follows the canonical nomenclature \((\mu + \lambda)\) [Beyer and Schwefel, 2002]. \(\mu\) is the number of parents that undergo mutations resulting in \(\lambda\) children. The “+” sign indicates that, when selecting the \(\mu\) “fittest” mutants, parents and children are taken into account. Since EAs are meta-heuristic optimization methods, they typically yield only approximately optimal solutions to a problem. The major advantage, however, is that they can handle discrete high-dimensional and multi-objective inputs and that their objective- or fitness function does not have to be (continuously) differentiable as in the case of, say, gradient descent based methods.

### III. METHODOLOGY

Next, we focus on our harvest scheduling problem. The question to be answered is if one can optimize the planting dates for a crop ensemble such that the harvest is efficiently distributed throughout the whole harvest season given all constraints. That is, the harvest is not supposed to overshoot a certain capacity but should be consistent throughout the season. Figure 1 sketches the problem. The next sections cover constraints, our initial evaluation criteria, and the mathematical problem formulation. Furthermore, we elucidate our adaptations to the optimization algorithm which are an oscillating mutation rate and the hierarchical loss.

#### A. CONSTRAINTS

Seeding and growing of crop are fraught with several constraints and uncertainties. Two constraints are that each crop species has individual windows or periods of possible planting. As a second constraint, a species has to be planted as a whole at one point in time. Furthermore, we introduce two variables. Firstly, the desired yield for each species and, secondly, the “performance” of the field to accumulate crop growth. In phenology, this is often measured by Growing Degree Units. This is the accumulated time-dependent temperature \(T(t)\) above a certain base temperature \(T_{\text{base}}\) during the interval of interest. Hence, \(\text{GDU} = \int (T(t) - T_{\text{base}}) \, dt\). Commonly, the interval is a day which is why it is also referred to as Growing Degree Days (GDD) [McMaster and Wilhelm, 1997]. Obviously, this GDU accumulation undergoes seasonal (and geographical) variations causing uncertainties on GDU forecasts. Since the forecast has direct impact on the predicted harvest time, uncertainties are propagated to it. Thus, the harvest time is a probability density over time points, although further simplifications like mean harvest time and standard deviations or a simple time interval are conceivable as well. We will optimize the harvest schedule with an EA which implies that we consider discrete time points and intervals. They do not necessarily have to be equidistant but one usually considers days or weeks as discretization units.

#### B. EVALUATION

In order to evaluate the quality of the optimization result, we consider six quantities deduced from the harvest schedule. The harvest schedule can be expressed as a vector \(\hat{h}\) whose elements denote the harvest yield at the corresponding date. The desired harvest yield, i.e. the target capacity, is called \(C_{\text{target}}\). Table 1 shows the evaluation criteria we will use in this work. As lower is better in each case, we model a minimization problem.

#### C. MODELING THE PROBLEM

Next, we introduce quantities that impact our problem. Let \(D = \{1, 2, \ldots, d_{\text{max}}\} \subset \mathbb{N}^+\) be the possible planting and harvesting dates. The dates themselves are encoded with an index \(d\) between 1 and the latest date \(d_{\text{max}}\). Recap, that the distance between dates does not have to be equal as long as planting and harvesting schedule are defined on \(D\).

Further, let \(S\) be the set of species. For each species, we have two dates in between which we could possibly plant the crop. The dates are now translated into the day index \(d\) from above. Thus, we have earliest and latest planting days for each species \(s \in S\) henceforth called \(d_{\text{early}}(s) \in D\) and \(d_{\text{late}}(s) \in D\).

Moreover, we have the required accumulated amount of GDU for each species \(s \in S\) to be harvested, i.e. \(g_{\text{harvest}}(s)\). The harvest date \(d_{\text{harvest}}(s, d_{\text{plant}})\) denotes the date when the sum of accumulated GDU reaches the required GDU value. Obviously, this date is dependent on the species \(s \in S\) and planting date \(d_{\text{plant}} \in D\). Hence, \(d_{\text{harvest}}(s, d_{\text{plant}})\) is determined by

\[
\min_d \left\{ d \geq d_{\text{plant}} \left| \sum_{x=d_{\text{plant}}}^{d} g_{\text{acc}}(x) \geq g_{\text{harvest}}(s) \right. \right\},
\]

by using the forecasted daily GDU accumulation function \(g_{\text{acc}}(x)\). Please note that we assume for the GDU accumulation that the plant gets the full accumulated GDU from the field on planting date as well as on harvesting date.
Given all this information, we can define a harvest matrix \( H \in D|S|\times |D| \) that contains the harvest date for each species (as rows) and each planting date (as columns). Outside the frame given by the earliest and latest possible planting dates, a species cannot be planted. We assign the value \(-1\) to those cases. Thus, the calculation instruction is \( H = \{H(i,j)\} \) with \( H(i,j) = \left\{ \begin{array}{ll} d_{\text{harvest}}(s = i, d_{\text{plant}} = j) & \text{if } j \in [d_{\text{early}}(i), d_{\text{late}}(i)] \\ -1 & \text{otherwise} \end{array} \right. \),
\( \text{(2)} \)

with harvest dates from Equation (1).

The benefit of having the harvest precalculated reduces the harvest schedule calculation to just a few lookups and a summation. This reduces the overhead in loss evaluation during runtime of the EA and, thus, accelerates our optimization. A second noteworthy fact is that the harvest matrix already includes all the constraint information of allowed planting windows for each species. Let \( \bar{q} = (q(s)) \in (\mathbb{R}|S|)^{|S|} \) be the vector of desired harvest quantity for each crop species. Then, one can calculate the weekly harvest \( \bar{h}(d_{\text{plant}}) \in (\mathbb{R}_{|D|}^+)^{|D|} \) by
\[
\bar{h}(d_{\text{plant}}) = \sum_{s \in S} \{I_{d_{\text{max}}}(H(s, d_{\text{plant}}(s)), d)q(s)\}_{d \in D},
\text{(3)}
\]
where \( I_{d_{\text{max}}} \) is the identity matrix with \( d_{\text{max}} \) dimensions. More descriptively, the harvest date index is looked up in the harvest matrix for a given planting date. The identity matrix maps the dates to the dates themselves again. Thus, the argument in the sum of Equation (3) is a one-hot vector with one element per date by evaluating the identity matrix for each possible date. Due to the single date planting constraint for each species, this vector represents in which week the corresponding yield is harvested. Summation over all species then results in the total harvest schedule vector.

Finally, our model takes the planting schedule for each species \( d_{\text{plant}}(s) \in D|S| \) as an input and calculates the harvest vector \( \bar{h} \) as the output. All constraints for the planting windows are included in the precalculated harvest matrix \( H \). The planting schedule has to be optimized. However, the goodness of optimization after all is not determined by the planting but by the harvest schedule. Furthermore, the date-to-date mapping done by the identity matrix can be adapted to a custom one, for instance a day-to-week mapping, as we will discuss later. Even more complex date patterns are thinkable, e.g. if harvesting should take place on a completely irregular number of days due to variable availability of logistical and human resources.

**D. DESIGNING THE OPTIMIZATION ALGORITHM**

In this work, we make use of an EA approach that uses mutation and selection, also referred to as \((\mu + \lambda)\) evolution strategy (ES). In each iteration (or generation), it takes \( \mu \) state vectors (parents) \( \overline{X} = (x_1, x_2, \ldots, x_n) \in \mathbb{R}^n \) with \( n \) individual states \( x_i \) and generates \( \lambda \) mutations (children). The \( \mu \) children and parents whose loss (or fitness) values are the best among the \( \mu + \lambda \) populations are considered to be the parents for the next generation. In this work, we consider the \((1 + 1)\)-ES, namely mutate one child per generation out of one parent. In our experimental evaluation, this turns out to be a good choice. In general, there is no optimal choice due to the No-free-Lunch-Theorem for Search [Wolpert and Macready, 1996]. In the case that parent and child have the same loss value, we prefer the child to the parent. In other words, we increase the probability to overcome plateaus in the loss landscape.

1) Oscillating Mutation Rate

The mutation is done by drawing from a probability distribution among all eligible states for each element. We decide to draw the new state from a uniform distribution \( U \). Each plant species has an individual and limited time window of possible planting days. Hence, \( d_{\text{plant}}(s) \sim U([d_{\text{early}}(s), d_{\text{late}}(s)]) \forall s \in S \).

In this work, we will use an initial mutation rate of \( 1/n \), so we expect one planting date to change in the whole schedule during one mutation step. After this initial step, we exploit the advantages of an adaptive mutation rate [Blum et al., 2005], [Agapie, 2001] by introducing a counter \( j = 0 \) that is increased by 1 if the mutant is worse than the parent. If the mutant is better than the parent, the counter is reset to 0. Moreover, there is the mentioned “plateau” case where the mutant is as good as the parent. In this case, the counter keeps its current value. Hence, we formulate an oscillating mutation rate
\[
\rho(j) = \frac{1}{|S|}(1 + (\rho_{\text{max}}|S| - 1) \sin^{2}(\omega j)),
\text{(4)}
\]

It describes an oscillating rate between \( \frac{1}{2|S|} \) and \( \rho_{\text{max}} \) with the frequency \( \omega \). Thus, we always have a mutation rate where at least one element and at most a ratio of \( \rho_{\text{max}} \) elements is expected to be mutated. The oscillating mutation rate enables the algorithm to try a wide range of mutations. At first, low mutation rates lead to very similar mutants with which the ES can scan the near region in the loss landscape for better solutions. If there are none, the increasing mutation rate makes the algorithm more flexible to overcome possible plateaus or local optima. The later decrease prevents the process from turning into a random walk. The reset to \( j = 0 \) causes a scan in the near region before scanning wider regions again. This principle is connected with the concept of “temperature” in simulated annealing [Kirkpatrick et al., 1983] but with advantage to be adaptive to the landscape in vicinity of the current best solution.

**E. HIERARCHICAL LOSS FUNCTION**

We will formulate the fitness via a loss function that has to be minimized. Hence, lower loss values correspond to a higher fitness. Let us consider the harvest quantities as a general ensemble of values. The evaluation criteria listed in Table 1 are only partly usable for a loss function of whole ensembles of values like the harvest quantity. Ensemble measures like
maximum or median of all values are not well suited as we would get lots of plateaus in the loss landscape resulting in an inefficient optimization. Therefore, we define a new objective that simply implements: “The harvest quantity should be equal to the capacity limit in as few as possible weeks.” For the whole ensemble, that means that all elements should optimally be either 0 or the target capacity limit. Note that this formulation of the objective will also minimize the previously given objectives.

Let $\vec{h} = (h(d))_{d \in D}$ be the vector of harvest quantities for each date and $C_{\text{target}}$ the target capacity limit. We now discriminate between two cases with different optimization goals. For $h(d) < C_{\text{target}}$, we want to have harvest either at capacity limit or no harvest at all, whereas for $h(d) \geq C_{\text{target}}$, we want to have as low overshoot as possible. Moreover, the total harvest is constant. Thus, if a mutation of the planting schedule results in the decrease of the harvest quantity at one date, it simultaneously increases the quantity at another date by the same value. Let us assume this particular case. We further assume that both dates’ harvest is below capacity limit. The loss should now promote high-harvest dates getting lower, and we hence define the loss function $l_C(h)$ as a convex loss function, whereas for harvest weeks above capacity, an exponential function acts as a convex loss function.

So, all in all, a concave loss function below capacity and a convex loss function above is desired. We define the loss vector $\vec{L}(h) = (L_+, L_-) = \left( \sum_{h \in h_+} l_C(h), \sum_{h \in h_-} l_C(h) \right)$ with

$$l_C(h) = \begin{cases} \frac{h}{C} (1 - \frac{h}{C}) & \frac{h}{C} < 1 \\ \exp(\frac{h}{C}) - \exp(1) & \frac{h}{C} \geq 1 \end{cases}, \quad (5)$$

where the vector $h_+$ only contains harvest quantities above and equal to the capacity limit and $h_-$ only contains those below. A plot of the loss function $l_C(h)$ is shown in Figure 2. The loss values are calculated by summation over all weeks belonging to the corresponding regions. As a concave part for the below capacity region $h/C < 1$, we use a concave polynomial, whereas for the above capacity region $h/C \geq 1$, an exponential function represents the convex part. The subtraction of $\exp(1)$ from the exponential function removes the unnecessary offset, so that $l_C(h = C) = 0$. In addition, considering the regions independently solves the problem when the change of harvest quantities happens from one region into the other. Otherwise, the net loss increase or decrease would depend on the scale or, respectively, the slope of the two different function segments.

Please note that these choices of concave and convex functions are just exemplary instances at this point. If a problem comes with the need to further adjust those functions, e.g. by availability of prior knowledge, this is easily possible as long as the concavity/convexity characteristics remain. Moreover, this approach is not restricted to two regions as one could add any number of additional criteria, even with overlapping domains. This enables a high adaptability to more complex problems.

Rather than optimizing both regions independently, resulting in a multi-objective approach, we still formulate a single-objective problem by preferring the loss in the region above capacity limit ($L_+$) over the region below it ($L_-$. Thus, every net change from over into under capacity is promoted. We call this approach a hierarchical loss. If, and only if, $L_+$ is constant during one mutation, $L_-$ is considered. If $L_-$ again gets less or equal by mutation, the mutation is the new best solution; if not, the mutation is discarded. Thus, the loss hierarchy goes down element-wise in the loss vector $\vec{L}$. One may add more loss categories in principle which makes this approach flexible for including further optimization criteria and knowledge. Algorithm 1 sketches our hierarchical comparison procedure with $n$ generalized hierarchical categories. Finally, the complete evolution and evaluation strategy is defined. The pseudocode is given in Algorithm 2.

All in all, the above mathematical formalization steps have the premise to reduce the calculation overhead during opti-

---

**Algorithm 1: Hierarchical comparison (HC)**

**Require**: $\vec{L} = (L_1, L_2, \ldots, L_n)$ (current best loss vector)

$\vec{L}^* = (L^*_1, L^*_2, \ldots, L^*_n)$ (loss vector of mutation)

1: $i \leftarrow 1$
2: while $i \leq n$ do
3: if $L^*_i \leq L_i$ then
4: if $L^*_i < L_i$ then
5: return $\vec{L}^* <_{HC} \vec{L}$
7: $i \leftarrow i + 1$
8: end if
9: else
10: return $\vec{L}^* >_{HC} \vec{L}$
11: end if
12: end while
13: return $\vec{L}^* = \vec{L}$

---

![FIGURE 2: Loss function (Equation (5)). For harvest weeks below the capacity limit (red dashed line), a polynomial acts as a concave loss function, whereas for harvest weeks above capacity, an exponential function acts as a convex loss function.](image-url)
Algorithm 2: Evolutionary optimization

Require: $d_{s,\text{ub}}, d_{s,\text{lb}}, H$
1: $j \leftarrow 0$
2: $\bar{d} \leftarrow \text{drawRandomPlantingSchedule}$
3: $\bar{d} \leftarrow \text{calculateHarvestSchedule}(\bar{d})$ [Eq. (3) or (6)]
4: $\bar{L} \leftarrow \text{calculateLossVector}(\bar{H})$ [Eq. (5)]
5: repeat
6: $\rho \leftarrow \frac{1}{\sqrt{2\pi}} (1 + (\mu|S| - 1) \sin^2(\omega j))$ [Eq. (4)]
7: $d^\star \leftarrow \text{mutation}(\bar{d}, d_{s,\text{ub}}, d_{s,\text{lb}}, \rho)$
8: $\bar{H} \leftarrow \text{calculateHarvestSchedule}(d^\star)$
9: $\bar{L}^\star \leftarrow \text{calculateLossVector}(\bar{H}^\star)$
10: $\text{hierarchicalComparison}(\bar{L}^\star, \bar{L})$ [Alg. 1]
11: if $\bar{L}^\star > HC \bar{L}$ then
12: $j \leftarrow j + 1$
13: else if $\bar{L}^\star < HC \bar{L}$ then
14: $j \leftarrow 0$
15: $d \leftarrow d^\star$
16: else if $\bar{L}^\star = \bar{L}$ then
17: $d \leftarrow d^\star$
18: end if
19: until termination
20: return $d$

TABLE 2: Overview of given datasets for both sites. The entries show the amount or shape of the respective data. For each species, the desired yield is drawn from a Gaussian distribution $\mathcal{N}(\mu, \sigma)$.

| Data                        | Site 0                                           | Site 1                                           |
|-----------------------------|--------------------------------------------------|--------------------------------------------------|
| GDU accumulation years 2009–2019, days | years 2009–2019, days                            |
| # plant species ($|S|$)      | 1375                                             | 1194                                             |
| required GDUs               | 1375                                             | 1194                                             |
| planting windows            | $1375 \times 2$, days                            | $1194 \times 2$, days                           |
| desired yield per species   | $\sim \mathcal{N}(250, 100)$                    | $\sim \mathcal{N}(350, 150)$                    |

IV. APPLICATION

We now concretize the procedure by applying the elaborated evolutionary harvest optimization algorithm to a dataset which covers two basic scenarios, a favorable and a rather unfavorable one. While processing the above method step by step, we will perform several dataset specific steps. Finally, we present and discuss the quantitative results.

A. DATASET AND NOMENCLATURE

In the considered dataset, two harvesting sites 0 and 1 are available. We will perform the optimization along two scenarios S1 and S2, targeting different objectives. In S1, the sites have weekly harvest capacity limits, whereas in S2, the capacity is not given. Here, a reasonable capacity should be found during optimization. Since the data is available on a daily basis, it is useful to introduce an integer index for the days, similar to the already introduced date indices. Our considered optimization period has non-negative date indices. We have to forecast the GDU accumulation by using historical data which has negative date indices. As stated in the Notation and Background part, we use a GPR forecast model. Details on the GPR model construction as well as the dataset itself can be found in the supplementary material.

The GDU accumulation forecast comes along with uncertainties. Fortunately, GPR models have the property of not just extrapolating unseen regions but also giving the confidence of their predictions. The actual predicted value is a mean value with a Gaussian standard deviation. We can exploit this to introduce uncertainties due to deviations in GDU accumulation forecasts, i.e. weather anomalies. In order to estimate the sensitivity of our optimization to different GDU accumulation scenarios, i.e. weather conditions, we resample many harvest matrices by the bootstrap technique [Efron and Tibshirani, 1993] and evaluate our result for all. We further substitute the identity matrix in Equation (3) for date-to-date mapping with a day-to-week mapping binary week matrix $W \in \{0, 1\}^{D \times |W|}$. Hence, $W = \{W(i, j)\}$ with $W(i, j) = 1$, if $\frac{|i - s|}{7} = j$ and 0, otherwise. Given this, Equation (3) takes the form

$$
\bar{h}(d_{\text{plan}}) = \sum_{s \in S} \left\{ W(H(s, d_{\text{plan}}(s)), W) q(s) \right\}_w \in W,
$$

(6)

with the harvest matrix $H$ from Equation (2).

The mathematical model is now tailored to the concrete use case. The GDU accumulation for harvesting season 2020/21 is forecasted by using a historical dataset of 11 years.
 guarantees the lowest overall deviations and, thus, outperforms the reference methods. The hierarchical loss, given in the legend, lists the amount of available data for both sites.

FIGURE 6: Comparison of different optimization strategies. For S1 and site 1, we compare our method with two established EAs, namely NSGA-II (left) and MOEA/D (middle). In addition, the right plot shows a comparison with an exact method (MILP). In all plots, the lower the bars, the better the optimization state. For the evolutionary approaches, we show the 6-objective (blue bars) and the 2-objective (orange bars) optimization with the respective algorithm. Our hierarchical (1+1)-ES method, shown by the green bars in all plots, has the lowest overall deviations and, thus, outperforms the reference methods. The hierarchical loss, given in the legend, is only used for the 2-objective and our approaches.

| Site 0 | S1        | S2-1       | S2-2       |
|--------|-----------|------------|------------|
|         |           |            |            |
| $C_{\text{target}}$   | 7000.00   | 10800.78   | 8716.42    |
| $C_{\text{need}}$     | 8392.02   | 11644.06   | 11643.02   |
| $OR_{\text{max}}$    | 19.89%    | 7.81%      | 33.58%     |
| $OR_{\text{min}}$    | 6773.90   | 7020.78    | 7936.42    |
| $\Delta_{\text{max}}$ | 30.00    | 530.72     | 615.58     |
| $\Delta_{\text{min}}$ | 49          | 46         | 52         |
| $\pi_{>\text{C}}$   | (9.90, 0.41) | (3.82, 0.68) | (22.39, 0.49) |
| $C_{\text{target}}$   | 6000.00   | 8364.24    | 7481.79    |
| $C_{\text{need}}$     | 6000.00   | 8368.00    | 7948.00    |
| $OR_{\text{max}}$    | 0.00%     | 0.05%      | 6.20%      |
| $OR_{\text{min}}$    | 3895.00   | 17.24      | 6678.79    |
| $\Delta_{\text{max}}$ | 0.00      | 0.76       | 434.73     |
| $\Delta_{\text{min}}$ | 52          | 51         | 56         |
| $\pi_{>\text{C}}$   | (0.00, 0.34) | (0.01, 0.01) | (7.20, 0.29) |

TABLE 3: Benchmark values for all optimization scenarios after $1 \times 10^6$ generations. Rows 2 to 7 show the evaluation criteria defined in Table 1, the last rows show the values of our hierarchical loss objective.

for both planting sites. Figure 3 shows the result for both sites 0 and 1.

B. HARVEST SCHEDULE OPTIMIZATION

For the optimization scenarios, our proposed EA applies the adaptive mutation rate shown in Equation (4). We use $\rho_{\text{max}} = 1\%$ and $\omega = 5 \times 10^{-4}$. With the given frequency, one oscillation needs $\frac{\pi}{\omega} \approx 6300$ counter steps. Table 2 briefly lists the amount of available data for both sites.

Our hierarchical loss needs the capacity limit for the separation between the two regions. However, this target capacity limit is only given for scenario S1. In S2, we have to find a preferably small target capacity limit by the method. Determining the lowest capacity needed is always a trade-off with the number of total harvest weeks. Therefore, we split S2 into two slightly different approaches, S2-1 and S2-2. The optimization is exactly the same as for S1 but with different inference of the capacity limit. In S2-1, we link the target capacity limit $C_{S2-1} = C_{\text{target}}(S2-1)$ to the number of harvest weeks the current solution needs. For S2-2, we...
determine the maximum number of possible harvest weeks. Hence, \( C_{S2-1} = (\sum_{h \in h} h) \mid \{ w \in W_{\text{harvest}} \mid h(w) > 0 \} \) and
\( C_{S2-2} = (\sum_{h \in h} h) \mid W_{\text{harvest}} \). It implicitly is \( C_{S2-1} \geq C_{S2-2} \), i.e. we expect S2-1 to need a higher capacity in less weeks, whereas S2-2 needs a lower capacity in more weeks if not all possible weeks are harvest weeks after all.

In Figure 5, the weekly harvest quantity is plotted against the harvest weeks for site 1. The light yellow bars are the weekly harvest quantity if the plants are planted by the original planting schedule given in the data. Blue bars represent our optimized result after \( 1 \times 10^9 \) generations. Plots for site 0 can be found in the supplementary material.

The computation of our method was implemented using Python 3.9. We performed independent experimental runs with different initial states (seeds). Each \((1 + 1)\)-ES run was executed on a single Intel® Xeon® E5-2640 v3 CPU. For our experiments with site 0 and 1, this results in a computation speed of about 1200 to 1500 generations per second.

V. DISCUSSION
In the following, we illuminate our optimization results from a technical as well as a social impact point of view.

A. OPTIMIZATION PROCESS
All scenarios show that the main optimization process needs orders of \( 1 \times 10^6 \) generations to saturate at a moderately well optimized state. The optimization process plot in Figure 4 shows that behavior exemplarily for two scenarios and three independent runs each. More of those plots for other scenarios can be found in the supplementary material. Beyond \( 1 \times 10^7 \) generations, fine-tuning occurs, e.g. if a constellation is found that a complete harvest week can be canceled for. The optimization process directly shows the characteristics and the strength of our method. Firstly, the hierarchical character of the loss is observable by a monotonically decreasing \( L_+ \) and an \( L_- \) showing jumps to higher values. Secondly, fine-tuning preferably happens with higher mutation rates than the intermediate saturation. Using the adaptive mutation rate prevents the EA from getting stuck in local optima. Further sophisticated optimization needs stronger mutations, i.e. higher mutation rates.

The results for S1 on site 1 completely vanished the overshoot. On site 0, the result is satisfactory although some overshoot still exists. For S2, the mentioned trade-off between harvest dates and capacity limit is visible. The target capacity \( C_{S2-2} \) is lower than \( C_{S2-1} \). Thus, the overshoot percentage is much higher (cf. Table 3) which means that targeting lower capacity is quite useless. The capacity has to be significantly higher to cope with the exceeding weeks, anyway, and more harvest weeks are needed. Thus, we would strongly prefer to accept the higher capacity in S2-1 for this case.

1) Comparison between Many-, Multi-, and Hierarchical Single-objective Optimization
In order to show that transferring the harvest scheduling into a single-objective optimization problem improves the convergence, we test our \((1 + 1)\)-ES against the commonly known algorithms NSGA-II and MOEA/D as a baseline. To get a proxy for the speed of convergence, we feature how the methods perform in a short-run optimization stopped after \( 1 \times 10^6 \) seen populations. We compare a many-objective approach with the six suggested criteria stated in Table 1 with our hierarchical single-objective approach. Additionally, a multi-objective approach by using the two components of the loss vector \( L \) as (coordinative) objectives is evaluated. The right and middle plots of Figure 6 show the results for the three stated problem formulations. The goodness of optimization is shown by considering distance to the next optimum (capacity limit or zero harvest) for each harvest week. Thus, lower bars represent better optimization. As expected, our approach shows a faster convergence to a rather preferable interim solution. Certainly, the solutions can be further improved by a long-run optimization.

2) Comparison with Exact Methods
Indeed, it is possible to formulate our problem as a mixed-integer linear programming (MILP) problem. However, as stated in the Introduction chapter, exact methods like MILP need to have objective functions that are less generic than those of, for instance, (meta-)heuristic approaches like EAs. Thus, we have to deal with several limitations that can not be considered by the MILP method. For instance, the concave loss function cannot be minimized by the solver. Additionally, our scenarios with variable target capacity cannot be represented by an appropriate objective function either. The right plot in Figure 6 shows the result, if we solve the S1 scenario (fixed target capacity) of our harvest scheduling problem by using the commercial solver CPLEX®. As an objective function, we are limited to use the summed squared error over all weeks between harvest yield and target capacity. As well as for the common two compared EA methods, our method clearly outperforms the MILP method. Note that the given MILP solution is already the best solution we can get at this point while our method was stopped in an early state. This vividly shows that complex problems like harvest scheduling can be solved better if the used optimization method allows for including sophisticated objectives and prior knowledge.

B. SOCIAL IMPACT
It is certainly debatable whether the rather abstract scenarios in this work are meaningful examples from reality. However, our method is flexibly extendable to further different influencing factors, since the loss vector can include arbitrarily many elements. Thus, even more complex scenarios can be transferred to a comprehensible single-objective optimization problem.

Our defined loss vector components have illustrative interpretations. For instance, \( L_+ \) minimizes food or crop waste...
by overshoot reduction, whereas $L_\infty$ minimizes logistic costs accompanied by greenhouse gas emissions. By construction, our hierarchy prefers food waste prevention to emission reduction. Firstly, it reduces the food waste by accepting additional emission. Secondly, the emission is reduced by distributing the yield preferably efficient. Given the hierarchy, no new food waste is created thereby. The baseline comparison (cf. Figure 6) shows that this is not just an economically and ecologically desirable objective. It substantially accelerates convergence as well.

Furthermore, the bootstrap shows that the optimization just for one GDU forecast overfits on that single scenario. This is visible by the resulting error bars in Figure 5. Optimizing for many GDU forecast realizations is expensive due to many loss evaluations. Nevertheless, we observe that the uncertainty propagation from GDU forecast to harvest yield still stays within standard deviation. Thus, our optimization is robust against environmental forecast uncertainties and can schedule the harvest reliably. Since the optimization can be run infinitely long in principle, one can even think about re-optimizing the model, e.g. once a more precise forecast is available. The harvest matrix just needs to be updated and optimization continues from the latest state.

We introduce the overshoot (undershoot) reduction ratio $R_o$ ($R_u$) that describes to which extent we can reduce food waste and emissions. As a reference, we take the unoptimized case given in the data. The values are given in Figure 5 and show that we can reduce 62% to 100% of over- and undershoot for both sites and all scenarios which is a drastic improvement. Using our method, one could thus tackle the global food waste problem on a supply chain as well as on a farming level. In addition, this contributes to an even more efficient agricultural land use.

VI. CONCLUSION

In this work, we have elaborated a generic method to apply the established concept of evolutionary computation to harvest scheduling tasks. Our new hierarchical loss and the accompanied simplification of multi- or many-objective problem formulations into single-objective tasks enables optimizing comprehensible objectives. Multi- or many-objective optimizations have ambiguous solutions that could result in compromises and trade-offs for the final solution. With our pseudo-single-objective approach, we can directly distinguish worse solutions from better ones. Additionally, our oscillating mutation rate enables the optimization to escape from local optima and plateaus considering the dynamics of the loss landscape. With all those adaptations, we demonstrate that we can optimize rather generic harvest scheduling problems more efficiently than common EAs. Even if the formulation as a linear programming problem—that could be tackled with solvers like CPLEX® or GUROBI®—fails, our approach can take over. This creates space for the integration of non-linear, multi-faceted constraints and objectives that may be given in the large field of logistics and supply chain management. Further research may contain evaluations about parallelizing the proposed method by generalization of the $(1+1)$-ES into a $(\mu+\lambda)$-ES or even more advanced strategies that, for instance, also apply recombination steps.

ACKNOWLEDGEMENT

This work was partially funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany’s Excellence Strategy – EXC 2070 – 390732324. Other parts were funded by the Federal Ministry of Education and Research of Germany as part of the competence center for machine learning ML2R (01IS18038B).

REFERENCES

[Kirkpatrick et al., 1983] Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. P. Optimization by simulated annealing. Science, 220(4598):671–680, 1983. ISSN 0036-8075. .

[Holland, 1992] Holland, J. H. Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence. MIT Press, 1992.

[Efron and Tibshirani, 1993] Efron, B. and Tibshirani, R. An Introduction to the Bootstrap. 1993.

[Bäck, 1996] Bäck, T. Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms. Oxford University Press, 1996.

[Manikas and Cain, 2016] Manikas, T. and Cain, J. T. Genetic algorithms vs. simulated annealing: A comparison of approaches for solving the circuit partitioning problem. In Computer Science and Engineering Research 1. 1996.

[Wolpert and Macready, 1996] Wolpert, D. and Macready, W. No free lunch theorems for search. Technical report, Santa Fe Institute, 1996.

[McMaster and Wilhelm, 1997] McMaster, G. and Wilhelm, W. Growing degree-days: one equation, two interpretations. Agricultural and Forest Meteorology, 87:291–300, 1997.

[Agapie, 2001] Agapie, A. Theoretical analysis of mutation-adaptive evolutionary algorithms. Evolutionary Computation, 9(2):127–146, 2001. ISSN 1063-6560. .

[Beyer and Schwefel, 2002] Beyer, H.-G. and Schwefel, H.-P. Evolution strategies – A comprehensive introduction. Natural Computing, 1(1):3–52, 2002. ISSN 1567-7818. .

[Deb et al., 2002] Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T. A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evolutionary Computation, 6(2):182–197, 2002. .

[Blum et al., 2005] Blum, S., Puisa, R., and Wintermantel, M. Adaptive Mutation Strategies for Evolutionary Algorithms. 2nd Weimar Optimization and Stochastic Days, pages 1–13, 2005.

[Rasmussen and Williams, 2006] Rasmussen, C. E. and Williams, K. I. Gaussian Processes for Machine Learning. MIT Press, 2006.

[Yu and Leung, 2006] Yu, R. and Leung, P. Optimal partial harvesting schedule for aquaculture operations. Marine Resource Economics, 21(3):301–315, 2006. .

[Datta et al., 2007] Datta, D., Deb, K., Fonseca, C., Lobo, F., Condado, P., and Seixas, J. Multi-objective evolutionary algorithm for land-use management problem. Int. J. of Computational Intelligence Research, 3:371–384, 2007.

[Zhang and Li, 2007] Zhang, Q. and Li, H. MOEA/D: A multiobjective evolutionary algorithm based on decomposition. IEEE Trans. Evolutionary Computation, 11(6):712–731, 2007. .

[Branke et al., 2008] Branke, J., Deb, K., Miettinen, K., and Slowiński, R., editors. Multiobjective Optimization: Interactive and Evolutionary Approaches. Springer, 2008.

[Ferreir et al., 2008] Ferreir, J., Cavley, A., Maturana, S., Toloza, S., and Vera, J. R. An optimization approach for scheduling wine grape harvest operations. International Journal of Production Economics, 112:985–999, 2008.

[Adeyemo et al., 2010] Adeyemo, J., Bux, F., and Otieno, F. Differential evolution algorithm for crop planning: Single and multi-objective optimization model. Int. J. of Physical Sciences, 5:1592–1599, 2010.

[Bauckhage and Kersting, 2013] Bauckhage, C. and Kersting, K. Data Mining and Pattern Recognition in Agriculture. KI – Künstliche Intelligenz, 27(4):313–324, 2013.
[FAO, 2013] FAO. Food Wastage Footprint. Impact on Natural Resources. Food and Agriculture organization of the United Nations, 2013.

[Singh et al., 2016] Singh, A., Ganapathysubramanian, B., Singh, A. K., and Sarkar, S. Machine learning for high-throughput stress phenotyping in plants. *Trends in Plant Science*, 21(2):110–124, 2016. ISSN 1360-1385.

[Mohammed et al., 2017] Mohammed, M. A., Abd Ghani, M. K., Hamed, R. I., Mostafa, S. A., Ahmad, M. S., and Ibrahim, D. A. Solving vehicle routing problem by using improved genetic algorithm for optimal solution. *J. of Computational Science*, 21:255–262, 2017. ISSN 1877-7503.

[Yandun Narvaez et al., 2017] Yandun Narvaez, F., Reina, G., Torres-Torriti, M., Kantor, G., and Cheein, F. A. A survey of ranging and imaging techniques for precision agriculture phenotyping. *IEEE/ASME Transactions on Mechatronics*, 22(6):2428–2439, 2017.

[Dong et al., 2018] Dong, L., Bettinger, P., Qin, H., and Liu, Z. Reflections on the number of independent solutions for forest spatial harvest scheduling problems: A case of simulated annealing. *Silva Fennica*, 52, 2018.

[Kamilaris and Prenafeta-Boldú, 2018] Kamilaris, A. and Prenafeta-Boldú, F. X. Deep learning in agriculture: A survey. *Computers and Electronics in Agriculture*, 147:70–90, 2018. ISSN 0168-1699.

[Lin et al., 2018] Lin, Y., Alawieh, M. B., Ye, W., and Pan, D. Z. Machine learning for yield learning and optimization. In *Int. Test Conference (ITC)*. IEEE, 2018.

[Cheraghalipour et al., 2019] Cheraghalipour, A., Paydar, M. M., and Hajiaghi-Akbari-Kesheti, M. Designing and solving a bi-level model for rice supply chain using the evolutionary algorithms. *Computers and Electronics in Agriculture*, 162:651–668, 2019. ISSN 0168-1699.

[FAO, 2019] FAO. The State of Food and Agriculture 2019. Food and Agriculture organization of the United Nations, 2019.

[Hassanat et al., 2019] Hassanat, A. et al. Choosing mutation and crossover ratios for genetic algorithms—a review with a new dynamic approach. *Information*, 10(12):390, 2019. ISSN 2078-2489.

[Roghanian and Cheraghalipour, 2019] Roghanian, E. and Cheraghalipour, A. Addressing a set of meta-heuristics to solve a multi-objective model for closed-loop citrus supply chain considering CO2 emissions. *Journal of Cleaner Production*, 239:118081, 2019. ISSN 0959-6526.

[Santos et al., 2019] Santos, P. A. V. H. d., Silva, A. C. L. d., Arce, J. E., and Augustynczik, A. L. D. A mathematical model for the integrated optimization of harvest and transport scheduling of forest products. *Forests*, 10(12), 2019. ISSN 1999-4907.

[Storm et al., 2019] Storm, H., Baylis, K., and Heckelei, T. Machine learning in agricultural and applied economics. *European Review of Agricultural Economics*, 47(3):849–892, 2019. ISSN 0165-1587.

[Slowik and Kwasnicka, 2020] Slowik, A. and Kwasnicka, H. Evolutionary algorithms and their applications to engineering problems. *Neural Computing and Applications*, 32(16):12363–12379, 2020.

[Varas et al., 2020] Varas, M., Basso, F., Maturana, S., Osorio, D., and Pezoa, R. A multi-objective approach for supporting wine grape harvest operations. *Computers & Industrial Engineering*, 145:106497, 2020. ISSN 0360-8352.

[Monteiro et al., 2021] Monteiro, A., Santos, S., and Gonçalves, P. Precision agriculture for crop and livestock farming—brief review. *Animals*, 11(8):2345, 2021.

[Taşkın and Bilgen, 2021] Taşkın, T. and Bilgen, B. Optimization models for harvest and production planning in agrifood supply chain: A systematic review. *Logistics*, 5(3), 2021. ISSN 2305-6290.
VII. SUPPLEMENTARY MATERIAL

A. DETAILED DESCRIPTION OF THE DATASET

The given dataset was distributed by Syngenta within the scope of the Crop Challenge\(^1\) 2021 and contains daily GDU time series from 01/01/2009 to 12/31/2019 for two sites (named as site 0 and site 1). Original, earliest, and latest planting dates of 2569 crop species (1375 for site 0, 1194 for site 1) and the required GDUs for each of those are given. On this dataset, two scenarios are exercised with randomly drawn desired harvest quantities per species. They are sampled from Gaussian distributions \(\mathcal{N}(\mu, \sigma)\) with \(\mathcal{N}(250, 100)\) and \(\mathcal{N}(350, 150)\), respectively. For scenarios S1, the sites have weekly harvest capacity limits (7000 units for site 0, 6000 units for site 1) whereas for S2, the capacity is not given. The optimization is performed for harvest season 2020/2021.

Plant and harvest windows for each plant species and both sites are shown in Figure 7 colored in blue and orange, respectively, in the upper plots. For site 1, they are quite continuous but for site 0, one can see accumulations regarding possible planting days around February and May. As a result, one can see two accumulations for the harvest days around November and the following February. The plant and harvest day distribution for site 1 is considerably smoother, although accumulations can also be seen here. A deliberate choice of the individual yield per plant could equalize this effect. However, yields are roughly drawn from Gaussian distributions in our case. The lower plots in Figure 7 confirm that by showing that weighting the day distribution by corresponding yield per species is not flattening the overall shape—neither for scenario S1 nor for S2.

B. GDU ACCUMULATION FORECAST WITH GPR MODEL

A naive approach to forecast the GDU accumulation would be to group the historical data by calendar date—i.e. month and day—and average over years. One would get an estimate for the daily jitter (or noise) by additional calculation of the standard deviation. Finally, the forecast could be a sampling of the future calendar days by a Gaussian distribution with the calculated means and standard deviations. However, this ansatz has some disadvantages. On the one hand, we have a problem when it comes to leap years. There is no clear assignment of “leap-year days” to “non-leap-year days”. However, this effect has not a severe impact, which is why we would just drop the leap days from the dataset. A more important drawback, on the other hand, is that this method is blind for possible underlying seasonalities or trends. This gets obvious if you think about trends due to climate change or seasonalities beyond annual scale, as—for instance—the El Niño phenomenon. Averaging over years would make those effects invisible or only noticeable by an increase of the standard deviations. This is, why we prefer to forecast the GDU accumulation by using a GPR model.

GPR models use kernels as priors for data prediction, inter-, and extrapolation. If we choose those kernels deliberately, we can include our knowledge about seasonality into the GPR process. We will choose a kernel that is a sum over the following fundamental kernels:

- a periodic exponential kernel with annual periodicity to represent the obvious annual seasonality,
- a linear kernel to represent possible large-scale trends over the historic period (for instance climate change),
- a bias kernel to cope with the fact that the GDU values are not centered around zero, and
- a white noise kernel to estimate random daily temperature fluctuations.

The forecast results for the GDU accumulation functions are shown in Figure 8 for both sites. Although there are some anomalies at some days, one cannot see any other periodic behavior in the residue plots in the two figures that would not have been caught by the GPR model. Thus, we can state that the GPR model covers the given dataset quite well. Due to the periodicity, we can simply extrapolate into the future (i.e.

---

\(^1\)https://www.ideaconnection.com/syngenta-crop-challenge/
The result for both sites is shown in Figure 3. All in all, the GDU data of site 1 has a higher volatility than site 0 which is reflected in the overall higher standard deviation compared to site 0.

![GP regression for site 0](image)

**FIGURE 8:** GPR prediction result for site 0 and 1. In order to get a more compact optimization domain, the day indices are normalized to years. In the upper plot, the data is shown as well as the mean prediction result and the confidence band with represents ±2σ in terms of Gaussian standard deviation. The lower plot shows the residues with respect to mean with the standard deviation of prediction as error bars.

where the 52 arises from the number of weeks per year. This means that one considers the same seeding over multiple seasons, i.e. repeating harvest plans can also be optimized. Figure 11 shows an exemplary optimization result of site 1 and scenario S2-1. Especially in this case, the overshoot is reduced to below 0.2 % with respect to the capacity limit. The optimization process is shown in Figure 12.

**C. HARVEST SCHEDULE RESULTS FOR SITE 0**

Figures 9 shows the optimization process for scenario S1 and site 0. The different colors represent 3 independent runs. The two lower plots show the components of the loss vector \( L_+ \) and \( L_- \) against number of iterations. The upper plot shows the mutation rate for each successful improvement as a dot.

![Optimization site 0, scenario 1](image)

**FIGURE 9:** Optimization process for scenario S1 and site 0. The different colors represent 3 independent runs. The two lower plots show the components of the loss vector \( L_+ \) and \( L_- \) against number of iterations. The upper plot shows the mutation rate for each successful improvement as a dot.

In addition to a single harvest time window, it is possible to define a day-to-week mapping for a cyclic harvest season

\[
W_{\text{cyclic}} \in \{0; 1\}^{\lceil D \rceil \times 52}
\]

by

\[
W_{\text{cyclic}} = \{W_{\text{cyclic}}(i, j)\},
\]

\[
W_{\text{cyclic}}(i, j) = \begin{cases} 
1 & \left\lceil \frac{d - d_0}{365.25} \right\rceil \mod 52 = j \\
0 & \text{otherwise}
\end{cases}
\]

(7)
FIGURE 10: Weekly harvest quantity for all 3 scenarios on site 0. The orange bars symbolize the harvest per week for an unoptimized schedule given in the dataset. Blue bars show our optimization result. The dashed lines mark $C_{\text{target}}$ (red) and $C_{\text{need}}$ (purple). For the uncertainty estimation, plotted as error bars, 100 harvest matrices are drawn from the GPR model for the GDU accumulation functions.

FIGURE 11: Weekly harvest quantity for scenario S2-1 and site 1 with cyclic harvest weeks. In the legend, all benchmark values are stated (cf. Table 3). For the uncertainty estimation, plotted as error bars, 100 harvest matrices are drawn from the GPR model.

FIGURE 12: Optimization process for scenario S1-1 and site 1 with cyclic harvest weeks. The different colors represent 3 independent runs. The two lower plots show the components of the loss vector $L_+$ and $L_-$ against number of iterations. The upper plot shows the mutation rate for each successful improvement as a dot.