OPTIMIZATION OF IMPORT OF ASSETS WITHIN A NETWORK

Wouter Zwinkels
5840007

10 july 2025

TU Delft Department of mathematics

Supervisor: Dr.ir. T.M.L. Janssen

Abstract

To improve the sustainability of electrical energy in the world, solar panels and windmills are introduced all over the world. However, since the actual weather differs from the forecast, the energy produced can differ from the predicted amount of energy. This introduces energy trading. This thesis looks at how to control the import and export of energy from different devices (called assets) within a single location, to minimize the total energy costs.
All assets at a location are grouped into (multiple) so-called EANs, each with their own energy contracts and prices. The total energy cost is made up of three parts:

- The deviation costs, which is the cost of an asset using more or less energy than its given optimum;
- The energy supplier cost, paid over the total amount of energy imported or exported per EAN;
- The transport cost, paid over all imported energy from the net.

When the deviation costs are linear, a linear programming (LP) method can be used to find the optimal solution to minimize the energy costs. However, the time complexity can be improved. Therefore, a fitting algorithm is made, which also results in the optimal solution. This algorithm works for linear deviation costs and even for piecewise linear deviation costs as long as they are convex.
The problem becomes more complex when the deviation costs are piecewise linear, but non-convex. Then finding the optimal solution is proven to be NP-Hard. Although it is very complex, a fitting algorithm can be made that finds solutions within 1.57% of the optimum. This results in an average cost reduction of 48.61% compared to the unoptimized case, where each asset operates independently.
Finally, if the deviation costs are non-linear, but still convex, a similar algorithm can be found as the algorithm for the problem with piecewise linear deviation, but this algorithm takes longer as it can be seen as a piecewise linear function with infinite intervals; however, this scenario is not very realistic currently.
In conclusion, the energy distribution problem can be optimally solved if the deviation cost only increases or if they do not change at all. If the deviation cost also decreases compared to its prior deviation cost, a solution can be found very close to the optimal solution.

Laymen's summary

To improve the sustainability of electrical energy in the world, solar panels and windmills are introduced all over the world. However, since the actual weather differs from the forecast, the energy produced can differ from the predicted amount of energy. This introduces energy trading. This thesis looks at how to control the import and export of energy from different devices (called assets) within a single location, to minimize the total energy costs.
At a location, for example, a company, different assets can be active. Some of these assets can be controlled, such as solar panels or batteries. Each of these assets has a given individual optimal import in kW and a deviation cost in €/kW, that is, the cost of deviating from this optimum. If all of these assets were looked at individually, they would logically all be at their optimum. However, if there are multiple assets on a location, it might be profitable to deviate from an asset optimum if this benefits the total energy cost.
All assets at a location are grouped into (multiple) so-called EANs, each with their own energy contracts and prices. The total cost consists of three parts:

- The deviation costs, which is the cost of an asset using more or less energy than its given optimum;
- The energy supplier cost, paid over the total amount of energy going in or out of an EAN;
- The transport cost, over all imported energy from the net.

The goal is to minimize the sum of these costs. In the base case, the deviation costs are linear; every asset has a fixed deviation cost in €/kW. This situation can be easily solved with a linear programming (LP) method, or a fitting algorithm can be made, which is faster if a lot of assets are present.
If the deviation costs change every once in a while, it becomes more difficult to solve. If the deviation cost only increases compared to the previous cost, the same algorithm can be used again, and it remains quite simple. However, if this is not the case, the problem becomes very difficult and thus quick algorithms cannot always find the optimal solution. A quick algorithm can still be made with results within 1.57% of the optimal cost, which is an improvement of 48.61% over the unoptimized scenario.
If the deviation cost can change at every moment (infinitely many), while still increasing, a quick algorithm can still be made.
In conclusion, the energy distribution problem can be optimally solved if the deviation cost only increases or if they do not change at all. If the deviation cost also decreases compared to its prior deviation cost, a solution can be found very close to the optimal solution.

Contents

Introduction

The energy transition is one of the most important aspects in the battle against global warming. Previously, the most important energy source was from fossil fuels, but since this creates a lot of CO2 emissions, a different energy source had to be used. This led to the introduction of solar panels, windmills, and other electrical energy-generating devices. This is a more durable energy source, as it does not produce a lot of CO2 emissions. However, there is still a problem with this method. If more or less energy is produced than expected, there will be a (negative) surplus of energy on the net. Therefore, the network operator will give money to the companies that can correct this difference. If there is a surplus of energy on the net, a company can buy new energy from the net very cheaply, and if there is a shortage of energy on the net, companies can get a high price for supplying energy to the net. This creates a new market; energy trading. This thesis will research how to find the optimal import and export of energy of all controllable assets within an energy network. This will be referred to as the energy distribution problem.

The optimization takes place at a specific location. This location can be a factory or some other company with a connection to the electricity network. Every location has a contract which limits the maximum import and export of energy; the throttle. This contract also determines the transport cost, i.e., a tax over the energy imported from the net.
At this location, multiple assets are connected. This can be a solar panel, a battery, a charging station, or a machine that consumes energy. Some of these assets can be controlled. A solar panel or a charging station can be turned on and off, and the import of a battery can also be controlled. These are the assets that will be looked at mainly.
Each of these assets is also in a certain EAN (European Article Numbering). An EAN can be seen as a grouping of assets. Each asset is part of exactly one EAN. Sometimes there is only one EAN, then all assets on the location are in this EAN. Each EAN has an energy contract with certain costs known as the energy supplier costs. This determines the price paid over the import and export of electricity on this EAN. When the contract is fixed, this price does not fluctuate very often; however, when this contract is not fixed, these prices can change every fifteen minutes.

In this thesis, the control of the import of controllable assets will be discussed. The goal is to manage them in such a way that the total energy costs are kept to a minimum. This will be done every minute, since the parameters can change quickly. In the first chapter, the base problem will be introduced. The parameters will be explained, and two solutions to the problem are given. In Chapter 2, we will look at the effect of changing the deviation cost to a piecewise linear function. Finally, in Chapter 3, the effect of non-linear deviation costs will be discussed, and an algorithm to find the optimal solution will be looked at.

Chapter 1

The Energy Distribution Problem

The goal of the energy distribution problem is to minimize the total costs. The total energy cost consists of the following costs.

- Deviation costs per asset (€/kW)
- Energy supplier costs per EAN (€/kW)
- Transport cost on location level (€/kW)

The deviating cost is the cost of an asset that deviates from its optimal import. This optimal import is based on the current net prices or, in the case of batteries, on a prediction of the future net prices. The total cost of the deviation per asset is the deviation cost multiplied by the difference between the actual import and the optimal import in kW. In Figure 1.1, an example of the deviating cost of multiple assets is shown.

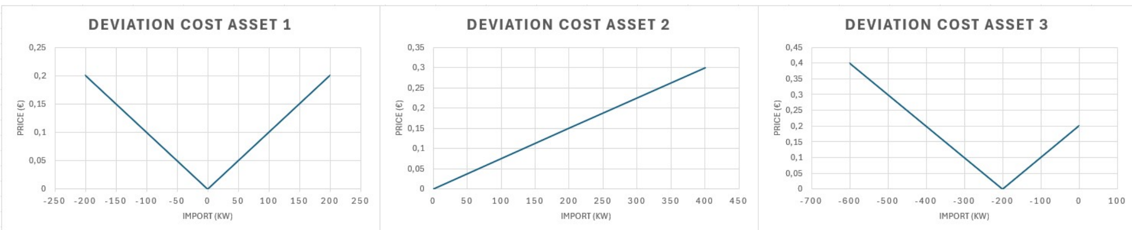


Figure 1.1: Deviation costs in different assets

The energy supplier costs are calculated per EAN. Each EAN has its own contract that determines the energy supplier cost of this EAN. The total cost per EAN is the amount of energy flowing in or out of the EAN in kW, multiplied by its energy supplier cost. An example of the total energy supplier costs is given in Figure 1.2. Finally, the transport cost is calculated over the amount of energy imported from the net. This transport cost depends on the network operator. When energy is only exported to the net, there is no transport cost. The total cost for this is the transport cost multiplied by the amount of energy imported in kW. An example of the total transport costs is given in Figure 1.2.

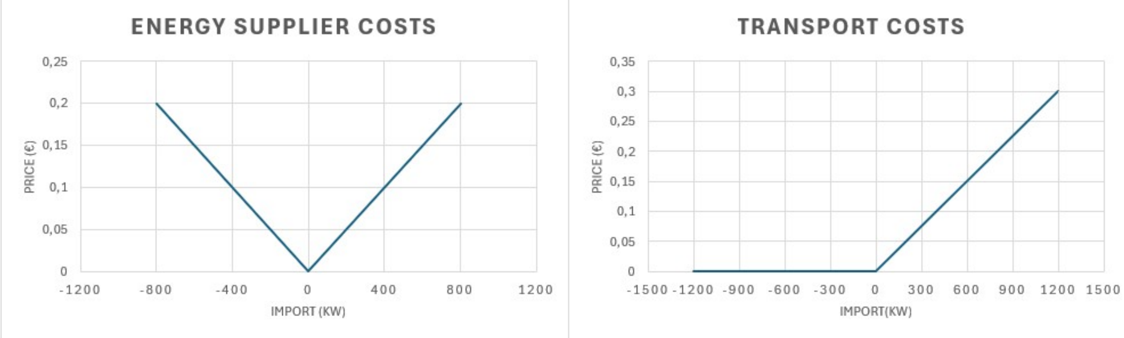


Figure 1.2: Example of Energy Supplier costs (left) and Transport Cost (right)

The total costs will therefore be defined by equation 1.1.

$$\sum_{i=1}^{n} |x_i - opt_i| \cdot dev\_cost_i + max(0, \sum_{i=1}^{n} x_i) \cdot trans\_cost + \sum_{j=1}^{m} es\_cost_j \cdot | \sum_{i \in EAN_j} x_i| \quad (1.1)$$

where variables are defined as follows:

- x_i is the import of asset i in kW.
- opt_i is the optimum import of asset i in kW.
- EAN_j is a set that contains all assets that are in EAN j.
- dev_cost_i is the deviation cost of asset i in (€/kW)
- $trans\_cost$ is the transport cost on location level in (€/kW)
- es_cost_j is the energy supplier costs of EAN j in (€/kW)

1.1 LP optimization

To find the optimal x_i for each asset i a linear programming (LP) solver can be used. This LP solver will find the optimal solution given certain decision variables and constraints for these variables. The LP used in this project is Python PuLp [J.S25]. This application uses CBC to solve its problems [FRV+24].
Since the cost function found in Equation 1.1 is not linear, some additional decision variables must be introduced besides x_i. All decision variables are listed below, each with their own constraints.

1. x_i is the amount of import of asset i in kW.
   - $x_i \geq min_i$
   - $x_i \leq max_i$
2. $deviation_i$ is the absolute value of the deviation in kW.
   - $deviation_i \geq x_i - opt_i$
   - $deviation_i \geq opt_i - x_i$
3. $total\_import$ is the amount of positive import in kW.
   - $total\_import \geq 0$

- $total\_import \geq \sum_{i=1}^{n} x_i$

4. EAN_import_j is the absolute amount of import over EAN j in kW.
   - $EAN\_import_j \geq \sum_{i \in EAN_j} x_i$
   - $EAN\_import_j \geq -\sum_{i \in EAN_j} x_i$

Here, min_i and max_i are the minimum and maximum possible import for asset i. Two additional constraints must be added to ensure that the total import at the location is within the throttle limits.

1. $\sum_{i=1}^{n} x_i \geq min\_import$
2. $\sum_{i=1}^{n} x_i \leq max\_import$

Here, min_import and max_import are the minimum and maximum allowed import at the location.

Given these decision variables and constraints, the linear minimization function can be defined as in equation 1.2.

$$\sum_{i=1}^{n} deviation_i \cdot dev\_cost_i + total\_import \cdot trans\_cost + \sum_{j=1}^{m} EAN\_import_j \cdot es\_cost_j \quad (1.2)$$

This function can be optimized using an LP, where x_i is the optimal import for asset i when looking on location level. Since this is an LP, the time complexity of this method is at least $O^*(n^{2.055})$ according to Jiang (2020) [JSWZ20]. Here O^* is the notation of the time complexity where the polynomial factors are omitted. For example, $O(n^2 + 5n) = O^*(n^2)$. This is not bad, but it can be improved.

1.2 Algorithmic optimization

Another way to solve the energy distribution problem is by using a fitting algorithm. Let all assets start with its import equal to its optimal import. Each controllable asset can import less and/or more than it is currently doing. The cost of changing this import is the sum of the deviation cost, the energy supplier cost, and the transport cost. However, this sum can also be negative. For example, if we lower the amount of import from an asset, the total import of the EAN will also decrease, which can result in lower energy supplier costs. If the gain from energy supplier costs is higher than the loss due to the deviation costs, it is profitable to lower the import of this certain asset. This can be done in a more organized way.

Let us look at a location with 3 EAN's with each 3 assets. Each of these assets can have a deviation interval to the 'left' by importing less than currently and to the 'right' by importing more than currently. We can put all intervals of an EAN in a set; L for 'left' and R for 'right'. These sets can be sorted by deviation cost, per EAN, and per direction.
We now use these sorted intervals to find the optimal distribution of the energy for every possible import of the EAN. This is as follows. All assets start at their optimum. First, we look at deviating to the left. We take the cheapest interval and use this to deviate along the interval. The cost per kW in this interval depends on the deviating costs and the energy supplier costs. If the import gets closer to

zero, the energy supplier cost gets subtracted, if the import goes away from zero, the energy supplier cost gets added. This is done for all intervals to the left, in order of deviating cost, and next it is done for all intervals to the right, in the same order. This will create a piecewise linear function. This piecewise linear function represents the lowest possible cost for every possible amount of energy imported.

An example can be found in Figure 1.3. This EAN contains asset 1, a solar panel, asset 2, a loading station, and asset 3, a battery. The combined optimum import of these assets is -200 kW. If we look at importing less than currently, we see that asset 2 is the cheapest option. The 100 kW interval of asset 2 will be used and the cost is the sum of the deviation costs and the energy supplier cost (shown in the middle of Figure 1.4), since more energy flows out of the EAN. The next cheapest (and only) interval is in asset 3. This asset can deviate 300 kW to the left. There are no more intervals to the left, so we now look at the right. The cheapest interval there is again asset 2. The 150 kW interval is used, but now the energy supplier cost is subtracted from the deviation costs, since the absolute import is decreasing. Therefore, this interval has a negative cost. The next cheapest interval is from asset 1. This asset can deviate 400 kW to the right. We first get an interval of 50 kW to the right, whose costs are equal to the deviating costs minus the energy supplier costs. However, the zero mark has now passed, so the left 350 kW will have a significantly higher cost, since this means that the energy supplier costs will now be added to the deviation costs, since the absolute import is now increasing. The last interval is 100 kW to the right of asset 3 with even higher costs. A piecewise-linear convex function has now been created which can be used to find the optimal import of this EAN. This function is shown on the right in Figure 1.4.

Figure 1.3: Deviation cost of different assets

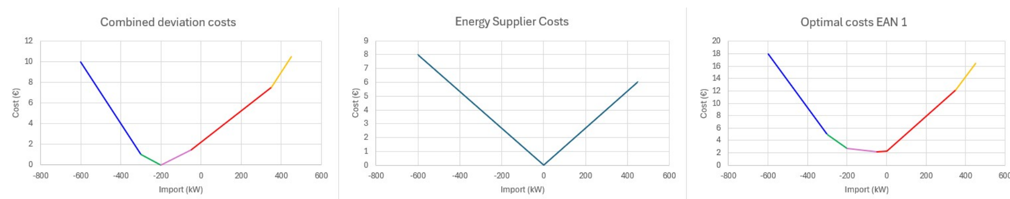


Figure 1.4: Combining the deviating costs (left) with the energy supplier costs (middle) gives the total optimum costs (right).

Let this be done for all three EAN's. Then we have three piecewise linear convex functions as in Figure 1.5. We can now create a similar piecewise linear function on location level by using the same algorithm as in the previous part. All intervals from the EAN function are sorted by their cost (starting at their minimum, left and right still separated), and since the functions are convex, we can use them in this order. We now combine this list of intervals with the transport cost, left in Figure 1.6, to obtain the piecewise linear function for the entire location, right in Figure 1.6. The minimum of this function is the minimal cost, and by bookkeeping which intervals have been used to get to this minimum, we can then find the new optimal import for each asset. Note that the function is only defined on $[-1200\text{kW}, 1200\text{kW}]$ since these are the boundaries of the location, and more import or export is not allowed.

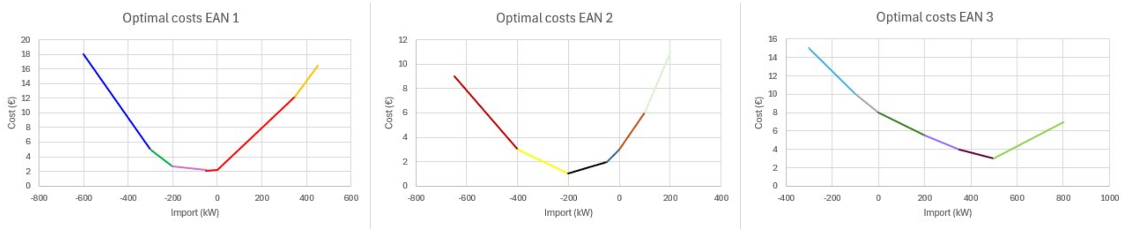


Figure 1.5: Optimal cost of different EAN's

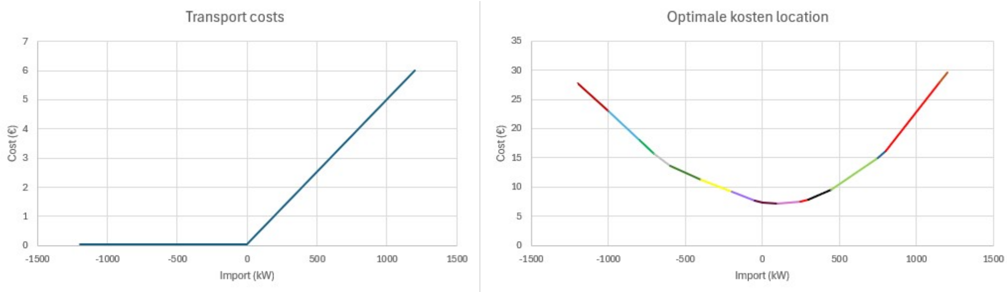


Figure 1.6: Combining the transport costs (left) with the optimal EAN costs (Figure 1.5) gives the total optimal location costs (right).

A pseudocode of this algorithm can be found in Algorithm 1. First, the empty sets L and R are created. Then, all intervals for both directions, l and r, are created and placed in the correct set, L and R, respectively. These intervals each consist of a length and a cost *interval_cost*. Then, these sets are sorted by their cost. The algorithm starts at the combined optimum of the parts. Then it uses the intervals in order, combining its cost with positive or negative energy supplier costs. This loops until we run out of intervals or the location boundaries are reached. This results in a function with the cost and optimal distribution of the import over the assets for every amount of total import.

This algorithm can first be used to create a piecewise linear function for each EAN and finally, these piecewise linear functions can then be put in this same algorithm, with EANs instead of assets and transport cost instead of energy supplier cost, to find the piecewise function for the location. Using this, the optimum can be found. In the pseudocode, $PARTS$ is a list of assets or EAN's with a (piecewise) linear function.

Since *sort* is the largest operation that occurs in this algorithm, the time complexity of this algorithm is the same as the time complexity of the sorting. According to Patel (2024) [Pat24], Merge Sort is the quickest sorting algorithm and has a time complexity $O(n \cdot log\, n)$. This is therefore a faster method than using the LP optimization, since the time complexity of this method is $O(n^2)$. (n is here equal to the number of assets.)

Algorithm 1 The minimization algorithm

```
L ← ∅
R ← ∅
starting_point ← 0
for part ∈ PARTS do
    for direction ∈ (l, r) do
        Find intervals of part: part in direction: direction
        Add intervals to L or R
    end for
    starting_point = starting_point + part_optimum
end for
Sort L and R by cost per kW
ean_point_list ← ∅
Add starting_point with its cost to point_list
for direction ∈ (l, r) do
    current_point ← starting_point
    i ← 0
    while current_point is within boundary of direction do
        interval ← i^th item of L or R
        interval_cost = interval_cost ± ES_cost (or trans_cost)
        new_point = current_point + interval
        Add new_point to ean_point_list
        current_point = new_point
        i = i + 1
    end while
end for
```

The testing is done with assets with a random optimum between 20kW and 200kW, and deviation costs between 0.01€/kW and 1.00€/kW. The energy supplier and transport costs are also randomly generated between 0.01€/kW and 1.00€/kW. Multiple different scenarios have been tested, the results of which can be found in Table 1.1. In the column 'Assets' the distribution of the assets can be found. Each 'x' is an asset with a random optimum and deviation cost, and a white space is a new EAN. For example, 'xxx xx x' means that the first EAN has three assets, the second EAN has two assets, and the third EAN has one asset. The energy supplier cost and transport cost are also random In the columns 'Alg' and 'Unopt' the average total cost of five hundred iterations of each scenario is noted if, respectively, the above described algorithm is used to find a solution or the system is not optimized. Finally, in the last column, the average improvement in cost of the algorithm case compared to the unoptimized case is noted.

Results of the base algorithm				
Scenario	Assets	Alg	Unopt	Alg/Unopt
1	xxx xx	303.40	551.18	44.96%
2	xxxx xx	319.06	579.85	44.96%
3	xxx xxx xxx	438.66	823.63	46.75%
4	xxxxx xxxxx	342.03	710.12	51.82%
5	xx xx xx xx xx	644.96	1110.39	41.91%

Table 1.1: Results of the base algorithm versus the unoptimized scenario

In the table, it can be seen that the average improvement is 46.08%, which is very significant. This saves a lot of costs and can thus be very useful for companies with multiple assets.

Chapter 2

Piecewise linear deviation

In the previous chapter, the deviation cost in the energy distribution problem is linear and symmetric for all assets. However, this is not always realistic. In case of a battery for example, a little more import than the optimum will not be a big problem, but a lot of extra import might be a bigger problem, since this can result in a fully loaded battery, which can cause problems with importing in the future. Therefore, (limited) piecewise linear deviation costs might be relevant. Due to the complexity, this can be divided into two categories; convex and non-convex.

2.1 Convex piecewise linear deviation

Convex piecewise linear deviation brings a lot less complications along than non-convex piecewise linear deviation. This is because the second interval of the deviation is by definition more expensive than the first, so the algorithm can still sort all intervals by cost and operate in that order. Therefore, the algorithm for convex piecewise linear deviation is the same as the algorithm for assets with fully linear deviation. The only difference is that there are now multiple deviation intervals per asset and per direction, which can be used separately, whereas the original algorithm would use the entire deviation interval of an asset before moving on to another asset. This change only affects steps 6 and 7 in Algorithm 1. Here, multiple intervals can be found instead of only one interval.
Let us look at an example of this energy distribution problem with convex piecewise linear deviation. Similarly to the example in Chapter 1.2, asset 1 is a solar panel, asset 2 is a loading station, and asset 3 is a battery. However, the deviation costs are now piecewise linear, as can be seen in Figure 2.1. The algorithm will still sort the set of all intervals in both directions and will use this to determine the cheapest intervals. In this case, deviating to the left will start with using 100 kW from asset 3. Then it will again use an interval of 100 kW from asset 3. Next, it will use (in order): 50 kW from asset 2, 100 kW from asset 3, and finally 50 kW from asset 2. Deviating to the right will start with 200 kW from asset 1, and then continue with (in order): again 200 kW from asset 1, 100 kW from asset 2, 100 kW from asset 3, 100 kW from asset 1, and finally 50 kW from asset 2. The final result when combining the deviation costs with the energy supplier costs can be seen in Figure 2.2.

The piecewise linear deviating costs give more switches between the assets when deviating over the whole EAN, however, this does not bring any complications. The algorithm will only take slightly longer, depending on the amount of pieces in the function. The time complexity of the algorithm is still $O(n \cdot log\, n)$, but now n is the total number of intervals of all assets combined.

Figure 2.1: Piecewise linear deviation costs of different assets

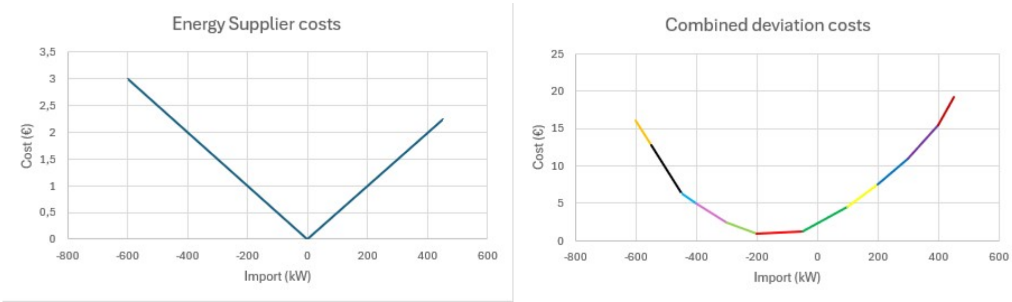


Figure 2.2: Combining the energy supplier costs (left) with the optimal EAN costs (Figure 2.1) gives the total optimal EAN costs (right).

2.2 Non-convex piecewise linear deviation

When the deviation is not convex, finding the optimal import for each asset can become quite complicated. This is because the algorithm must take into account that it might be profitable to use an expensive interval if the interval behind it is very cheap. Therefore, we can no longer sort all intervals and simply use the cheapest one.

To find the optimal solution, the algorithm must know the cheapest possible import distribution for every amount of import. This number of combinations can be very large since the energy can be distributed over all assets in many different ways. Therefore, an algorithm would take a long time to find this cheapest combination, making finding the optimal solution quite difficult.

Finding an optimal solution is actually so complex that it can be proven to be NP-Hard.

Theorem 1. *The energy distribution problem with non-convex piecewise linear deviation is NP-Hard.*

Proof. This will be done by reducing the subset sum problem to the energy distribution problem. In the subset sum problem, a set of integers $\mathcal{S}$ and an objective value V are given. The task is to find a subset $S \subseteq \mathcal{S}$, such that $\sum_{s \in S} s = V$.

When looking at the energy distribution problem with convex piecewise linear deviation, a similar case can be found. Let there be an arbitrary amount of assets, but

each with similar deviation costs; the cost of the first 1% of the total allowed deviation length is equal to 0.099€/kW $\cdot$ length, where *length* is the total length in kW and the second 99% of the total interval costs only 0.001€/kW$\cdot$*length*. For example, if an asset can import 100 kW more than its optimum, the deviating cost for the first kW is $9,90$€, and the total cost of deviating from one kW to one hundred kW extra only costs $0,10$€. Now, let the optimum for the assets in the EAN combined be equal to a value W. If the energy supplier cost were equal to $(0.10 + \alpha)$€/kW with $\alpha > 0$ significantly small enough, the whole interval must be used to make a profit by deviating (assuming the deviation is rounded to one decimal). To maximize profit, we must find a set of assets such that the sum of the total deviation of these assets is equal to W. This would create a profit of $W \cdot \alpha$, which is maximal.
We can now reduce the subset sum problem to this case. For every item $s \in \mathcal{S}$, let us create an asset with $[0, s]$ as its only interval, with cost as described above. Now, let V be equal to W. If the energy distribution problem with non-convex piecewise deviation is not NP-hard, we could find the optimal solution for this problem in polynomial time. All assets in the optimal solution will use a deviation of either 0 or s. Thus, the profit per asset is either 0 or $s \cdot \alpha$. If the optimal profit is $W \cdot \alpha$, we found a subset S of the assets such that $\sum_{s \in S} s = V$. If the optimal profit is less than $W \cdot \alpha$, there is no such subset since our solution is optimal. This, however, would imply that the subset sum problem is also not NP-Hard. This leads to a contradiction, and therefore the energy distribution problem with non-convex piecewise deviation is NP-Hard.

□

Although this is quite a difficult problem, a fitting algorithm can still be created to optimize the costs.
We start by looking at the current import on location level. Using this, we can create a possible interval length for both directions while not changing in transport cost. For example, if the current import is above zero, the length of the interval to the left is equal to the difference between the current import and zero, and the length of the interval to the right is equal to the difference between the current import and the boundary on the right. This way, the throttle cannot be exceeded. This will be referred to as the usable length in both directions.
Using this, we can look for a deviation in one of the assets that can be used to make a profit on this interval as follows. Two lists are created for possible deviation intervals. For each asset, the intervals are made by walking over the usable length and adding an interval with its length and cost to the list for each breakpoint (a point where the cost per kW of deviation changes) that is come across. Here, a change in the energy supplier costs also counts as a breakpoint. The interval over the entire usable length is also added to the list. When this is done, the energy supplier costs are added or subtracted, depending on the current state of the EAN. The result of this process is a list with the intervals of every asset for each direction.
The next step is to find the cheapest interval in both directions by sorting the lists and adding or subtracting the transport cost to the interval costs. Finally, if the cost of one of these intervals is below zero (if both are profitable, then the lowest is used), then this interval is used to deviate from the asset and the process repeats itself. This happens until no more profitable intervals can be found.

In Algorithm 2, a pseudocode for this algorithm can be found. First, $Current\_opt$ is found. Then, using this $Current\_opt$, the range is calculated in both directions. For each EAN, for each asset, the intervals to each breakpoint within this range are found in both directions, the energy supplier cost ES_cost is added or subtracted to the cost of these intervals and then the intervals are added to L or R. This list is then sorted. The cheapest intervals of both lists are taken and the transport cost is added. Note that $trans\_cost$ can be positive, negative, or zero, depending on $Current\_opt$ and the direction of $used\_interval$. Then the cheapest of these two intervals is taken, and this will be our $used\_interval$. If the cost of this interval is less than zero, then $used\_interval$ is added to $Current\_opt$ and the process repeats itself. If not, the algorithm is finished.

Algorithm 2 The minimization algorithm for non-convex piecewise deviation

```
Current_opt = Σ_{asset ∈ Asset_list} asset_optimum
Profitable = True
while Profitable = True do
    if Current_opt < 0 then
        Left_Range = Left_Boundary − Current_opt
        Right_Range = −Current_opt
    else if Current_opt > 0 then
        Left_Range = −Current_opt
        Right_Range = Right_Boundary − Current_opt
    else
        Left_Range = Left_Boundary
        Right_Range = Right_Boundary
    end if
    L = ∅
    R = ∅
    for direction ∈ (l, r) do
        for EAN ∈ EAN_list do
            for Asset ∈ EAN do
                Find interval for each breakpoint within Left/Right_Range
                Add or subtract ES_cost to interval_cost
                Add each interval to L or R
            end for
        end for
    end for
    Sort L and R on cost per kW
    for direction ∈ (l, r) do
        Cheapest_(direction) = the first interval in L or R
        Add trans_cost to Cheapest_(direction)
    end for
    used_interval = min(Cheapest_l, Cheapest_r
    if Cost of used_interval < 0 then
        Add used_interval to Current_opt
    else
        Profitable = False
    end if
end while
```

The results of this algorithm can be tested against an integer linear program (ILP). This ILP finds the optimal solution, but it takes a long time to calculate this when the assets have more deviation intervals. The testing is done with assets with random intervals of lengths ranging from 20kW to 200kW, and costs between 0.01€/kW and 1.00€/kW. The energy supplier and transport costs are also randomly generated between 0.01€/kW and 1.00€/kW. Multiple different scenarios have been tested, the results of which can be found in Table 2.1. In the column 'Assets', the distribution of the assets can be found. Each 'x' is a random asset, and a white space is a new EAN. For example, 'xxx xx x' means the first EAN has three assets, the second EAN has two assets, and the third EAN has one asset. In the columns 'Alg' and 'ILP', the average total cost of one hundred iterations of each scenario is noted if, respectively, the above described algorithm or the ILP is used to find a solution. In the column 'Alg/Unopt', the average improvement of algorithm cost compared to the unoptimized cost is noted. Finally, in the last column, the average improvement of the ILP cost compared to the unoptimized cost is noted.

Results Non-Convex Algorithm					
Scenario	Assets	Alg	ILP	Alg/Unopt	ILP/Unopt
1	xxx xx	293.29	285.84	50.46%	51.73%
2	xxxx xx	312.35	304.75	49.54%	50.80%
3	xxx xxx xxx	479.62	465.96	49.73%	51.14%
4	xxxxx xxxxx	375.83	365.43	51.92%	53.30%
5	xx xx xx xx xx	706.80	687.95	42.42%	43.95%

Table 2.1: Results of the non-convex algorithm versus the ILP

The average improvement over the unoptimized scenario is 48.61%, which is quite a good result, as the ILP has an average improvement of 50.18% over the unoptimized scenario. The algorithm is therefore only 1.57% worse than the optimal solution.
The most time-consuming step in the algorithm is again sorting, which happens once in every iteration of the while loop. The number of iterations of the while loop is bounded by the total number of intervals. Therefore, the time complexity of this algorithm is also $O(n^2 \cdot log\, n)$, where n is equal to the total number of intervals of all assets combined. This is a nice time complexity for such a good result.

Chapter 3

Non-linear deviation

The solution of this energy distribution problem also becomes more complex when the deviating costs are non-linear, even while assuming that they are convex. In the algorithm used in Section 1.2, the deviation intervals are sorted by their slope. However, if the deviation is non-linear, this slope changes constantly. Therefore, this algorithm must be adjusted to deal with the non-linear deviation.

The algorithm must start with the deviation with the lowest slope at the beginning of the deviation. It will continue to use this deviation as the cheapest deviation until it reaches a slope that is equal to the next lowest slope. This moment will be determined by comparing the derivatives of the deviation cost. When this point is found, the algorithm stops deviating from the first asset and continues with the second cheapest asset. It continues using the second asset until it reaches a steeper slope than the other options, and then it switches again. Using this method, the result will be a new non-linear function. If the functions of two (or more) assets are similar, it is possible that the algorithm will switch a lot between these assets; however, this is not a problem for the final result. To prevent infinitely many switches, a minimum length must be introduced in the algorithm. This ensures that the algorithm will use at least this length when it switches to an asset, which puts a lower bound on the number of switches.
Algorithm 1 is still quite relevant except for a few steps. Steps 6 and 7 must be changed so that the initial slope is added to the list instead of an interval. Now, when the while loop is reached, the slope is added to the current point and this point is added to the point list. The new slope of the used asset is then calculated and this slope is compared with the second lowest slope in the slope list. As long as this new slope is lower than the slope of the second cheapest asset, it will continue to add the slope to the points and calculate the new slope. If the slope of the second cheapest asset is lower, it will switch to this asset and will repeat this 'while' loop.

Let us look at a simple example. In Figure 3.1, the deviation costs of three assets can be found. Asset 1 has a deviation function: $dev\_cost = |x - opt|^2 \cdot 8 \cdot 10^{-5}$. Assets 2 and 3 have linear deviation costs, with a constant slope of 0.05 and 0.035, respectively. When comparing the derivatives of these functions in Figure 3.2a, we can see that asset 1 starts with the lowest derivative. Therefore, the algorithm will use this asset to deviate until it reaches a slope of 0.035. Then it will switch to asset 3, where the slope does not change. Once the maximum amount of deviation is used, the algorithm will continue to deviate from asset 1, until it reaches a slope of 0.05. Now, it uses the 100 kW from asset 2 to deviate. If this is done, it will finally use the last deviation interval from asset 1. The result will look like Figure 3.2b. This is a new non-linear function, which shows the cost of deviating over the whole EAN. This can be used to find the function for the cost per import of the whole location using the same algorithm, which will give the solution to the energy distribution problem.

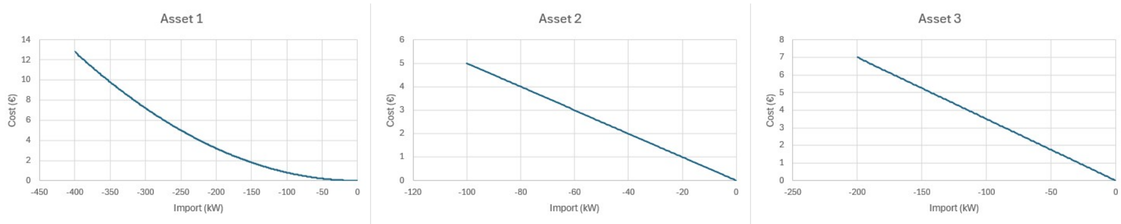
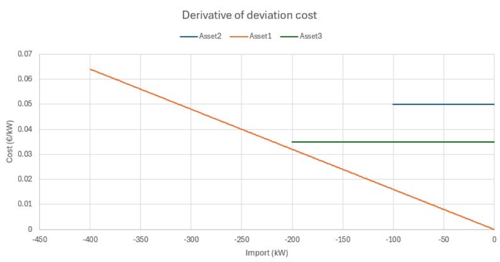
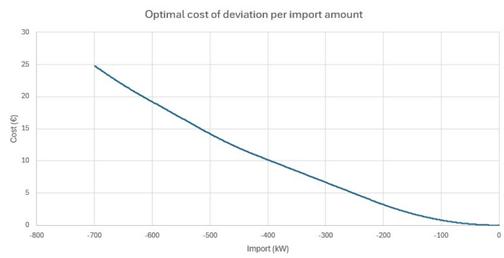


Figure 3.1: An example of (non-linear) deviation costs of different assets



(a) The derivatives of the deviation costs of different assets



(b) Total optimal cost per import on EAN level

Figure 3.2: Derivative of deviation costs (a) and the optimal cost per import (b)

Now, we can look at a more complex example. Figure 3.3 shows an example of non-linear deviation costs. In asset 1, the deviation cost function is defined as $dev\_cost = |x - opt|^3 \cdot 10^{-7}$. In asset 2, the deviation cost to the left is defined as $dev\_cost = (|x-opt|^2 + 16 \cdot |x-opt|) \cdot 10^{-3}$, while the deviation to the right is defined as $dev\_cost = (4 \cdot |x-opt|^3 + 10 \cdot |x-opt|^2) \cdot 10^{-6}$. Finally, in asset 3 the deviation cost in both directions is defined as $dev\_cost = (|x-opt|^4 + 3 \cdot 10^6 \cdot |x-opt|) \cdot 10^{-9}$. When comparing the derivatives of these functions (only looking at deviating to the left), we can see that asset 1 starts with the lowest derivative in Figure 3.4a. Therefore, when deviating to the left, asset 1 will be used first. After a while, the deviation costs of asset 1 start to rise and will increase above asset 3. When this happens, the algorithm will switch to asset 3 for the next deviation. Now, the deviating costs of asset 3 will start to increase, and thus the algorithm will switch back to asset 1. After a while, asset 2 will be the lowest, and then the algorithm will constantly switch between each of the three assets. If asset 1 is maximally deviated, it will continue to switch between assets 2 and 3 until asset 3 is also maximally deviated, and then the last part will just be asset 2. In Figure 3.4b, the order of the deviation pieces can be found. Here, it is clearly visible that the asset used for deviation can switch a lot if this algorithm is used. In this example, the interval lengths have been minimized at 1 kW. If there was no minimum interval length, the algorithm would switch infinitely many times.

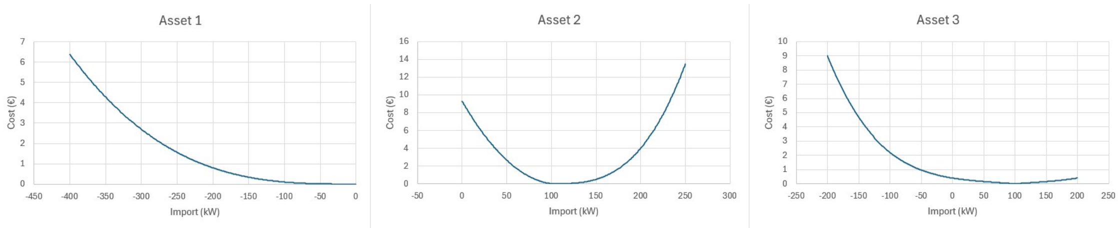
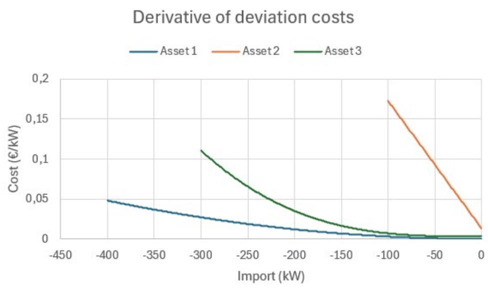
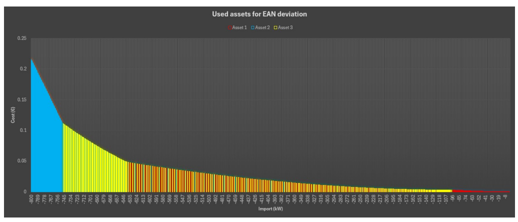


Figure 3.3: An example of non-linear deviation costs of different assets



(a) The derivatives of the deviation costs of different assets



(b) The optimal deviation per amount of import for the EAN

Figure 3.4: Derivatives of the deviation cost(a) and asset used for each deviation(b)

Chapter 4

Conclusion

The aim of this thesis was to minimize the total energy costs at locations with multiple assets and EAN's. This has been done for different situations; linear deviation costs, piecewise linear deviation costs, and non-linear deviation costs.

4.1 Results

In Chapter 2, the base problem was looked at. An LP could be used to solve this problem, but the time complexity, $O(n^{2.055})$, could be improved. Therefore, a fitting algorithm has been created to find the same optimal solution, but with a time complexity of only $O(n \cdot log(n))$. The improvement that has been made by finding the combined optimum, instead of individual optima, is 241.2% on average over five scenarios with each 500 iterations. This is a very significant improvement.
In Chapter 2, the deviation costs have changed from linear to piecewise linear. In the scenario where all deviation costs are still convex, the same algorithm as in Section 1.2 could be used, and thus similar improvements can be made by using this algorithm. The change from linear to piecewise linear will only ensure it takes slightly more time, since more intervals are used, and thus more iterations are needed.
If the piecewise linear deviation is non-convex, the algorithm is not usable any more, and a new solution must be found. This new problem can even be proven to be NP-Hard, as is done in Section 2.2. Nonetheless, an almost optimal solution can be found by creating a fitting algorithm, with a difference of only 1.57% from the optimal solution in randomly generated instances. This is very small, definitely compared to the improvement of 48.61% compared to the unoptimized scenario. This algorithm has a time complexity of $O(n^2 \cdot log\, n)$, which is quite good.
Finally, in Chapter 3, we have looked at a situation where the deviation costs of the assets are non-linear. In this situation, we can use an algorithm that deviates from the cheapest asset until another asset becomes cheaper. This is not a very complex algorithm, however, a lower bound for the interval lengths must be introduced to prevent the algorithm from constantly switching between the assets, as can be seen in the second example in the chapter. Although it is not really a realistic situation currently, this algorithm can still find an optimal solution.

4.2 Discussion

Improvements can be made to make the scenarios more realistic. For example, consumption tax is not included. This is a tax over the energy that is consumed by for example machines or loading stations, however this tax is not over the energy imported by batteries.
Another improvement that can be made is to take into account that the energy supplier costs and transport costs are calculated once every fifteen minutes, instead of every minute [Nex25]. This could mean that the throttle can be exceeded for a couple of minutes, as long as the average import over fifteen minutes does not exceed the throttle. Another possibility of improvement is by compensating for the previous minutes. For example, if the total import over an EAN is positive in the first few minutes of the quarter, then exporting the next few minutes could be profitable if the total import over the fifteen minutes gets closer to zero.
Also, state 2 regulations are not taken into account in this research [Nex25]. Simply explained, if there is a shortage of energy on the net over an interval of fifteen minutes, the price for exporting to the net is very profitable and importing from the net is quite expensive. If there is an overload of energy on the net, importing is very cheap, and exporting can cost money. However, if there is a shortage in the first ten minutes but an overload in the next five minutes, the cost of both importing and exporting is very high. Therefore, once state 2 regulations are noticed to be active, the average import per EAN should ideally be brought to zero (except if the deviation costs are even higher) to make sure these high energy supplier prices are evaded.

Other possible improvements could be to try different mathematical methods for solving the problems. For example, for the energy distribution problem with non-linear convex deviation, the gradient descent method [Kwi24] might be suitable to find a good solution quickly.
Since the algorithm for the energy distribution problem with non-convex piecewise deviation is not optimal, improvements can be made in that algorithms as well.

Finally, it is worthwhile to mention that the algorithms in this thesis are designed to solve the energy distribution problem, however, they may be applicable to other problems which require combining multiple functions to find a new function with a new optimum.

Acknowledgements

I would like to thank Dr.ir. T.M.L. Janssen for helping me write this thesis. He has had a great impact on the process of creating the algorithms.

Bibliography

[FRV+24] John Forrest, Ted Ralphs, Stefan Vigerske, Haroldo Gambini Santos, John Forrest, Lou Hafer, Bjarni Kristjansson, jpfasano, EdwinStraver, Jan-Willem, Miles Lubin, rlougee, a andre, jpgoncal1, Samuel Brito, h-i gassmann, Cristina, Matthew Saltzman, tosttost, and to st. coin-or/cbc: Release releases/2.10.12, 2024.

[J.S25] J.S.Roy. PuLP, 5 2025.

[JSWZ20] Shunhua Jiang, Zhao Song, Omri Weinstein, and Hengjie Zhang. Faster dynamic matrix inverse for faster LPs, 4 2020.

[Kwi24] Robert Kwiatkowski. Gradient Descent Algorithm — a deep dive - TDS Archive - Medium. *Medium*, 3 2024.

[Nex25] Regeltoestand 2 uitgelegd — NextEnergy, 1 2025.

[Pat24] Harshil Patel. Quicksort Algorithm: an overview, 9 2024.