**AIMMS Modeling Guide - Pooling Problem**

This file contains only one chapter of the book. For a free download of the complete book in pdf format, please visit www.aimms.com.

# Chapter 12

# A Pooling Problem

In this chapter you will encounter a simplified example of a refinery pooling problem. Intermediate product streams, each with their own particular properties, are fed into a limited number of available tanks. These tanks are referred to as pool tanks. Through pooling, the input products no longer exist in their original form, but are mixed to form new pooled products with new property values. These new products are subsequently used to blend final products. These final blends must satisfy specific quality requirements, which means that their property values must be between a priori specified ranges. The pooling problem can be translated into a nonlinear programming model. This model has the nasty property that there are likely to exist multiple locally optimal solutions. Good starting values are then required to steer the algorithm away from poor local optima. An instance of the pooling problem is provided for illustrative purposes.

*This chapter*

Pooling problems have received some attention in the informal literature of the sixties and seventies. A typical reference is [Ha78] which describes a heuristic approach based on recursive programming techniques. In later years, global optimization techniques have been proposed by [Fl89], but these have found limited application in large-scale practical applications.

*References*

Nonlinear Program, Multiple Optima, Worked Example.

*Keywords*

## 12.1 Problem description

In this section a simplified version of the pooling problem is discussed and illustrated. The paragraphs aim to give you a basic feel for the problem at hand. Despite its simplifications, the problem is still of interest, because it captures the difficulties associated with pooling problems in general.

*This section*

In a refinery, crude oils of different types are first *distilled* in one or more crude distillers. This process results in the production of several intermediate product streams that are immediately *pooled* in dedicated pool tanks. Any further processing consists of *blending* pooled products into final products. This three-step process is illustrated in Figure 12.1. In this chapter the time-

*Refinery operations summarized*

phasing of product flows is completely ignored in order to prevent the problem and the resulting model from becoming too complicated.
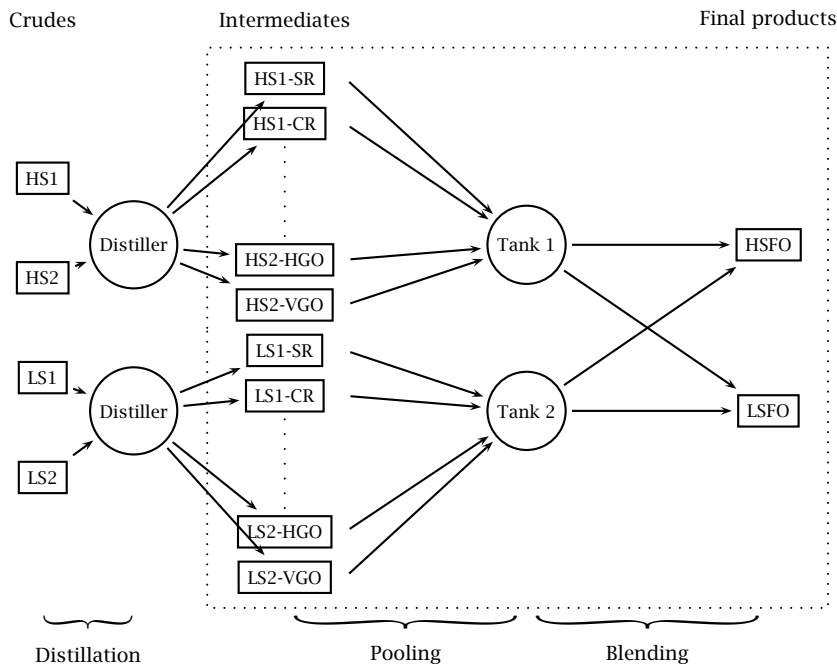


Figure 12.1: A simplified refinery

Crude oils are by no means identical. Their composition strongly depends on their location of origin. In Figure 12.1 there are four crudes: two of them are high-sulphur (HS) crudes and two of them low-sulphur (LS) crudes. Their sulphur content is referred to as a *product property*. Product properties are to a certain extent retained during the distillation phase. That is why the labels HS and LS are also attached to the intermediate product streams for each originating crude 1 and 2.

*Product properties...*

Product properties cannot be measured in a uniform manner. There are properties, such as sulphur content, that are naturally expressed as a percentage of total volume or total mass. Other properties, such as viscosity and pour point, are not naturally measured in these terms. For instance, pour point is the lowest temperature, expressed as a multiple of 3 degrees Celsius, at which oil is observed to flow when cooled and examined under prescribed conditions. Special care is then required to compute such a property for a mixture of products.

*... and their measurement*

When two or more intermediate products are pooled in a single tank, a new product will result. The properties of this new product will be related to the properties of the originating products, but there will always be some form of dilution of each property. When there are multiple pool tanks, it is desirable to minimize the dilution effect across pool tanks. The resulting variability in pool properties is needed to meet the property requirements of the final products. For instance, in Figure 12.1 all high-sulphur intermediates are not pooled with any low-sulphur intermediates in order to maintain sufficient variability in sulphur property values across the two pool tanks.

*Mixing properties causes dilution*

In Figure 12.1, each of the two crude distillers produces four intermediate products, namely, a short residue (SR), a cracked residue (CR), a heavy gas oil (HGO) and a visbroken gas oil (VGO). In order to track the properties of the originating four crude oils, the name of each intermediate is prefixed with the name of the crude. In this case, such a naming convention results in sixteen different product names. Only one final product, namely fuel oil (FO), is produced in two qualities, resulting in two final product names. You can imagine how the number of possible product names can explode in large real-world applications where there are more products and several additional properties. The number of product names becomes even larger when the entire refinery process is considered over several discrete time periods, and properties are tracked over time.

*Intermediate and final products*

Theoretically, it is attractive to store all intermediate products in their own intermediate product tanks. This delays any dilution of product properties until final products have to be blended to specification. In practice, however, the unique intermediate product streams outnumber the available tanks, and product pooling is required. For the sake of keeping the problem fairly simple, it is assumed that the flow of intermediate products into the pool tanks equals the flow required to blend the final products, and that each pool tank has limited capacity.

*Pooled products*

In the example of this chapter it is assumed that the volume of each intermediate product to be used must stay within specified bounds. This is a slight simplification, because in actuality, intermediate products are produced in fixed relative proportions, and their absolute volume is related to the volume of the emanating crude. However, the distillation phase is not part of the problem in this chapter, which justifies the assumption of fixed bounds on inputs.

*Limitation on intermediates*

The price of a final product is not entirely constant, but is assumed to be dependent on its associated property values. This implies that price becomes an unknown in the pooling model to be build. The objective function is to maximize the total sales value of final products to be made.

*Maximizing sales value*

The pooling problem considered in this chapter is to maximize the sales value of end products by deciding how much of each intermediate stream, within limits, is to be placed in each of the pool tanks. The extra restrictions are that the new pool mixtures are sufficient to produce the required amount of final products, and that the properties of the new mixtures are sufficient to satisfy final product requirements.

*Pooling problem summarized*

## 12.2 Model description

In this section the basic rules for blending on volume and blending on weight are developed before stating the mathematical formulation of the underlying pooling problem.

*This section*

For the sake of simplicity, consider two intermediate products (1 and 2) to be mixed into a new pooled product (3). Let the symbol $x$ denote the amount of product, and let the symbol $p$ denote a particular property. The following two equalities express proportional blending.

*Proportional blending . . .*

$$\begin{aligned} x_3 &= x_1 + x_2 \\ p_3 x_3 &= p_1 x_1 + p_2 x_2 \end{aligned}$$

The first identity is the product balance equation, which is linear. The second identity is the property determination equation, and is nonlinear when both $x_3$ and $p_3$ are considered to be unknown.

Proportional blending is properly defined when the units of measurement are consistent. Consistency is obtained, for instance, when product amounts are measured in terms of mass, and the product property is measured as a percentage of mass. Similarly, consistency is also obtained when product amounts are measured in terms of volume, and the product property is measured as a percentage of volume. If required, it is always possible to transform volume into mass or vice versa using product densities.

*. . . requires consistent measurements*

As has been mentioned in the previous section, there are several product properties, such as viscosity and pour point, that are not measured as a percentage of mass or volume. These properties play a vital role in determining the quality of a blend. In practice, a nonlinear function of such properties is constructed such that the resulting transformed property values can be viewed as a percentage of either volume or mass. The determination of such specialized nonlinear functions is based on laboratory experiments and curve fitting techniques.

*If this is not the case . . .*

Let $f(p)$ denote a nonlinear function of one of the properties discussed in the previous paragraph. Assume that $x$ is expressed in terms of mass, and that $f(p)$ is measured as a percentage of mass. Then the following identities express proportional blending.

*... then use transformed measurements*

$$x_3 = x_1 + x_2$$
$$f(p_3)x_3 = f(p_1)x_1 + f(p_2)x_2$$

You could of course use a variable for $f(p)$ and apply the inverse of $f$ to obtain $p$ after you have found the solution of the underlying model. This is what is typically done in practice.

By considering the basic pooling problem described in this chapter, it is fairly straightforward to describe the objective and constraints in a compact verbal manner.

*Verbal model statement*

**Maximize:**     *total sales value of final products*
**Subject to:**
- *for all pool tanks:  the bounded flow entering a pool tank must be equal to the flow leaving a pool tank,*
- *for all properties and pool tanks:  the property values of pooled product are determined by the property values of the products entering the pool tank,*
- *for all properties and final products:  the property values of final product are determined by the property values of the products coming from the pool tanks,*
- *for all final products:  the quantities of final product must be between specified bounds,*
- *for all properties and final products:  the property values of final product must be between specified bounds,*

The following notation is based as much as possible on the use of a single letter for each identifier for reasons of compactness. Such compact notation is not recommended for practical models built with a system such as Aimms, because short names do not contribute to the readability and maintainability of computerized models.

*Notation*

**Indices:**
| | |
|---|---|
| $p$ | *properties* |
| $i$ | *intermediates* |
| $t$ | *pool tanks* |
| $f$ | *final products* |

**Parameters:**
| | |
|---|---|
| $v_{pi}$ | *value of property $p$ in intermediate $i$* |
| $\underline{r}_i$ | *minimal amount of intermediate $i$ to be used* |
| $\overline{r}_i$ | *maximal amount of intermediate $i$ to be used* |

$$
\begin{array}{ll}
\underline{r}_f & \textit{minimal required amount of final product } f \\
\overline{r}_f & \textit{maximal required amount of final product } f \\
\underline{w}_{pf} & \textit{minimal value of property } p \textit{ in final product } f \\
\overline{w}_{pf} & \textit{maximal value of property } p \textit{ in final product } f \\
c_t & \textit{capacity of pool tank } t
\end{array}
$$

**Variables:**

$$
\begin{array}{ll}
v_{pt} & \textit{value of property } p \textit{ in pool tank } t \\
v_{pf} & \textit{value of property } p \textit{ in final product } f \\
x_{it} & \textit{flow of intermediate } i \textit{ to pool tank } t \\
x_{tf} & \textit{flow of pool tank } t \textit{ to final product } f \\
s_t & \textit{total stock of pooled product in pool tank } t \\
\pi_f & \textit{sales price of final product } f
\end{array}
$$

As discussed in the previous section, the amount of each intermediate product to be pooled is restricted from above and below. Instead of writing a single flow balance constraint for each pool tank, there are separate equations for both inflow and outflow using the same stock variable. It is then straightforward to specify a simple bound on the stock level in each pool tank. The following five constraints capture these limitations on flow from and to the pool tanks.

*Flow constraints*

$$
\sum_t x_{it} \ \geq \ \underline{r}_i \quad \forall i
$$

$$
\sum_t x_{it} \ \leq \ \overline{r}_i \quad \forall i
$$

$$
s_t \ = \ \sum_i x_{it} \quad \forall t
$$

$$
s_t \ = \ \sum_f x_{tf} \quad \forall t
$$

$$
s_t \ \leq \ c_t \quad \forall t
$$

The property value determination constraints are essentially the proportional blending equalities explained at the beginning of this section. These constraints are only specified for pooled and final products, because the property values of all intermediate products are assumed to be known.

*Property value determination constraints*

$$
v_{pt} \sum_i x_{it} \ = \ \sum_i v_{pi} x_{it} \quad \forall (p,t)
$$

$$
v_{pf} \sum_t x_{tf} \ = \ \sum_t v_{pt} x_{tf} \quad \forall (p,f)
$$

Due to market requirements with respect to quantity and quality, both the amount of final product and the associated property values must be between a priori specified bounds.

$$\sum_t x_{tf} \geq \underline{r}_f \quad \forall f$$
$$\sum_t x_{tf} \leq \overline{r}_f \quad \forall f$$
$$v_{pf} \geq \underline{w}_{pf} \quad \forall (p,f)$$
$$v_{pf} \leq \overline{w}_{pf} \quad \forall (p,f)$$

*Final product requirement constraints*

As has been indicated in the previous section, the price of a final product is not entirely constant, but is assumed to be dependent on its associated property values. In the specification below, only an abstract functional reference $F$ to property dependence is indicated. In the worked example of the next section a particular function is used for numerical computations. The objective function to be maximized can then be written as follows.

*Objective function*

$$\sum_f \pi_f \sum_t x_{tf}$$
$$\pi_f = F(v_{pf})$$

## 12.3  A worked example

In this section you will find a description of model input data that is consistent with the entities in Figure 12.1. In addition, you will find some comments based on running a few computational experiments with AIMMS.

*This section*

The variable $x_{it}$ denotes the flow of intermediate $i$ to pool tank $t$. In Figure 12.1, these intermediate flows are restricted such that all high and low sulphur products are pooled into separate pool tanks. In AIMMS, you can model this by simply specifying an index domain as part of the declaration of the variable $x$. Such a domain is then a parameter with nonzero entries for the allowed combinations of $i$ and $t$.

*Domain restrictions on x*

The symbol $v$ is used both as a parameter and a variable depending on the index references. In AIMMS, you can implement this dual status by declaring the symbol to be a variable, and then changing its status to non-variable for selected index combinations. You can accomplish this change by writing an assignment statement inside a procedure using the `NonVar` suffix. A typical assignment is

*Variable status of v*

```
v(p,i).NonVar := 1;
```

Both the lower and upper bound on the amount in [kton] of each intermedi- *Intermediate*
ate product to be pooled are displayed in Table 12.1. In this table you also *product data*
find the property values associated with the properties sulphur and V50, both
measured as a percentage of mass. The property V50 is a derived measure of
viscosity for which the proportional blending rule is appropriate.

| | $\underline{r}_i$ | $\overline{r}_i$ | $v_{pi}$ | |
| | | | Sulphur | V50 |
| | [kton] | [kton] | [%] | [%] |
|---|---|---|---|---|
| HS1-SR | 1 | 3 | 5.84 | 43.7 |
| HS1-CR | | 3 | 5.40 | 36.8 |
| HS1-HGO | | 3 | 0.24 | 12.8 |
| HS1-VGO | | 3 | 2.01 | 15.4 |
| HS2-SR | 1 | 3 | 5.85 | 47.3 |
| HS2-CR | | 3 | 5.38 | 39.2 |
| HS2-HGO | | 3 | 0.26 | 13.1 |
| HS2-VGO | | 3 | 2.04 | 15.9 |
| LS1-SR | 1 | 3 | 0.64 | 39.9 |
| LS1-CR | | 3 | 0.57 | 38.2 |
| LS1-HGO | | 3 | 0.02 | 13.5 |
| LS1-VGO | | 3 | 0.14 | 16.3 |
| LS2-SR | 1 | 3 | 0.93 | 38.1 |
| LS2-CR | | 3 | 0.85 | 34.1 |
| LS2-HGO | | 3 | 0.03 | 13.2 |
| LS2-VGO | | 3 | 0.26 | 15.5 |

Table 12.1: Intermediate product data

The only data solely linked to pool tanks is their capacity. In this simplified *Pool tank data*
example it is assumed that the capacity of each pool tank is 15 [kton].

In Table 12.1 you will find the upper and lower bounds on both the quantities *Final product*
and property values of the final products to be blended. *requirements*

| | $\underline{r}_f$ | $\overline{r}_f$ | $\underline{w}_{pf}$ | $\overline{w}_{pf}$ | $\underline{w}_{pf}$ | $\overline{w}_{pf}$ |
| | | | Sulphur | | V50 | |
| | Min | Max | Min | Max | Min | Max |
| | [kton] | [kton] | [%] | [%] | [%] | [%] |
|---|---|---|---|---|---|---|
| LSFO | 10 | 11 | | 1.5 | 30.0 | 34.0 |
| HSFO | 11 | 17 | | 3.5 | 32.0 | 40.0 |

Table 12.2: Final product requirements

In this particular example, the unit sales price of each final product is made dependent on its sulphur content, with low sulphur levels worth more than high sulphur levels. The base price of one [ton] of low sulphur fuel oil with the highest level of permitted sulphur is equal to 150 dollars. Similarly, the base price of one [ton] of high sulphur fuel oil with the highest level of permitted sulphur is equal to 100 dollars. These base prices can only increase as the relative level of sulphur decreases. The following formula for $\pi_f$ in terms of the base price $\pi_f^I$ is used.

*Price of final product*

$$\pi_f = \pi_f^I(2 - \frac{v_{\text{Sulphur},f}}{w_{\text{Sulphur},f}})$$

Once you have implemented the model of the previous section in AIMMS, you are likely to obtain an error indicating that "all Jacobian elements in the row are very small". This message comes directly from the solver, and is most likely a reflection of some initial variable values at their default value of zero. The algorithm uses a matrix with derivative values for all constraints in terms of all variables. Several of these derivative values correspond with the product of two variables. From basic calculus you know that the term $xy$ has zero partial derivatives with respect to both $x$ and $y$ when these are at their default value of zero.

*Initial model run*

The remedy to fix the problem mentioned in the previous paragraph is straight-forward. By initializing the flow variables $x$ away from their lower bound of zero, the error message will disappear. In this example you could consider random flow values for the intermediate products between their bounds, and distribute the corresponding content of the pool tanks equally over the final products. As a result, the flow balance constraint are already satisfied. As it turns out, specifying initial values for variables in nonlinear mathematical programs is just one of the ways to make the solution process more robust. Setting bounds on variables, and scaling your data such that solution values become of similar magnitude, are all useful ways to improve the likelihood that a solver will find a correct solution.

*Initialize flow values*

As you start to experiment with initial values for the $x$ variables, you might find that the solver still has difficulties finding a feasible solution in some of the cases that you try. As it turns out, you can reduce the number of times the solver is unable to find a feasible solution by also computing the initial values of the $v$ variables using both the values of $x$ and the property value determination equalities.

*Derive initial property values*

If you have not found different optimal solution values after experimenting with various initial values for $x$ and $v$, you may want to write an experiment in which you let the system generate random initial values for the $x$ variables and compute the corresponding values of $v$. It is easy to write such a procedure in AIMMS, and make a page to display the various objective function values in a table. In this example, two distinct local optimal objective function values were found. They are 3624.0 and 4714.4.

*Multiple solutions exist*

## 12.4 Summary

In this chapter a simplified pooling problem was introduced together with a small data set for computational experiments. The problem was transformed into a nonlinear programming model with proportional blending constraints to determine new product property values. Initial values of the solution variables were needed not only to support the solution finding process, but also to determine different locally optimal solutions.

## Exercises

12.1 Implement the mathematical program described in Section 12.2 using the example data provided in Section 12.3.

12.2 Experiment with several initial values by writing a procedure in AIMMS as described in Section 12.3. Try to find at least two different local optima.

12.3 Investigate the effect of removing all pool tanks on the objective function value. Without pool tanks, final products are blended on the basis of intermediate product streams only.

# Bibliography

[Fl89]  C.A. Floudas, A. Aggarwal, and A.R. Ciric, *Global optimum search for nonconvex NLP and MINLP problems*, Computers & Chemical Engineering **13** (1989).

[Ha78]  C.A. Haverly, *Studies of the behaviour of recursion for the pooling problem*, ACM SIGMAP Bulletin **26** (1978).