Robust real-time optimization for the linear oil blending

Stefan Janaqi, Jorge Aguilera, Mèriam Chèbre

Robust real-time optimization for the linear oil blending

Stefan Janaqi[a], Jorge Aguilera[a], Meriam Chèbre[b]

[a]LGI2P Ecole des Mines d'Alès, Parc Scientifique Georges Besse, 30035 Nîmes Cedex 1, France
[b]Advanced Process Control Department, Technical Direction, TOTAL Refining &Marketing, Le Havre, France

Abstract

In this paper we present a robust real-time optimization method for the online linear oil blending process. The blending process consists in determining the optimal mix of components so that the final product satisfies a set of specifications. We examine different sources of uncertainty inherent to the blending process and show how to address this uncertainty applying the Robust Optimization techniques. The polytopal structure of our problem permits a simplified robust approach. Our method is intended to avoid reblending and we measure its performance in terms of the blend quality giveaway and feedstocks prices. The difference between the nominal and the robust optimal values (the price of robustness) provides a benchmark for the cost of reblending which is difficult to estimate in practice. This new information can be used to adjust the level of conservatism in the model. We analyze the feasibility of a blend to be produced from a set of feedstocks when the heel of a previous blend is used in the composition of the new blend. Additional critical information for the control system is then produced.

Key words: Blending, Robustness, Polyhedra, RTO, Linear Programing.

1. Introduction

The oil blending process consist in determining the optimal proportions to blend from a set of available feedstocks or components such that the final product obtained fulfills a set of specifications on their properties. A blending system is typically constituted by three functional subsystems: the scheduling subsystem, the online optimizer and the control subsystem.

The *scheduling subsystem* is the one in charge of the general refinery production planning. The scheduling subsystem uses a linear program to calculate for a given period (up to a month) the recipes u, properties y and volumes V of reference for a sequence of blends $(u_k^0, y_k^0, V_k^0), k = 1, \dots, K$ (typically $K \in [10,15]$). These calculations are based on the mean characteristics of components.

Several sources of uncertainty (see below) perturb the process. The *online optimizer* is then required to update the target recipe which may became sub-optimal, or even infeasible, due to these disturbances in the process. For the first blends of the sequence, the online process fits well with the forecasted (u_k^0, y_k^0, V_k^0). Nevertheless, after a number of blends, it happens sometimes that the blending environment differs significantly from the mean characteristics considered by the scheduling subsystem. In these cases (the example presented at the end of the paper is one of them), it is very difficult for the online optimizer to take the right decision. The blending environment is represented in this paper by the blending polytopes (see section 2) that change with time and are different from mean polytopes used by scheduling subsystem.

The feedback is based on measures of the blends' and components' properties gathered by online analyzers. Finally, the *control subsystem* is in charge to adjust the component's flow rates (the recipe u) in order to attain the current target recipe. Here we focus on the Real Time Optimization (RTO) system formed by the online optimizer and the control subsystem. An RTO loop consist of calculating the optimal recipe u^* for the new constraints and adjusting the actual recipe u by $u^* - u$. Typically, our control subsystem adjusts the recipe every 5 minutes by calling the online optimizer with different sets of constraints (up to 100 calls in 5 minutes). The total time for a blend is several hours (10-20 hours). A RTO is one that meets this time constraint. This motivates our approach for robust blend models that can be calculated efficiently. Our uncertainty model gives rise to linear models that are efficiently solved.

An important characteristic in the oil blending process is the strict requirements over some properties controlled by environmental, legal and technological standards. If these properties are not satisfied, one can correct the blend to a certain limit by dumping the appropriate additives in it. Otherwise, the blend must be reblended. The additives are too expensive whereas reblending reduces the refinery capacity, so one should consider these blends' bounds as hard constraints which must be satisfied.

On the other hand, oil blending is a complex process where several unknown and uncertain factors affect the blend's properties. In addition to the plantmodel error produced by the linearization of blending laws, there are other sources of uncertainty: measure errors on components and properties caused by instruments' precision; uncertain knowledge of the components' properties due to upstream process variations and uncontrolled blending conditions such as air temperature, humidity, etc. All these are typical uncertainties arising in any RTO problem (see [6]).

Hard constraints on the blends' properties combined to these sources of uncertainty are the ideal characteristics to apply the Robust Optimization (RO) techniques (see [1]), which are an approach to optimize under uncertainty. Roughly speaking, robust optimization looks for optimal solutions that are strictly inside the optimization domain. The robust optimization models differ in the way they insure and measure the "how much inside the optimization domain?". A strength of the RO techniques is to consider a deterministic and set-based uncertainty model if one has no information on the random process(es). No probabilistic assumption is made over the uncertainty and the solution obtained is optimal for any realization of the uncertainty in a given set. RO techniques seem to be ad hoc to address an uncertain RTO problem where the feasibility is the primary concern.

Some previous works have been devoted to find robust solutions to the blending problem. In [3], a geometric approach is presented for the product and mixture design problems but only considering the uncertainty due to measurement imprecision. To deal with components' properties uncertainty, [4] proposed a nonlinear blend RTO system based on predictions of feedstocks properties whereas [5] presented a chance constraint model and a hybrid neural networks–genetic algorithm solution. More recently, [2] introduced a linear blending control algorithm which handles this type of uncertainties via an estimator of the components' properties. A more general method based on stochastic programming which covers various types of uncertainty is presented in [7].

Our contribution begins by highlighting the polytopes underlying the blending process. This and the form of uncertainty on components (see below) permit to conceive robust models without the need of convex programming tools. Based on the data, we develop robust models that give rise to linear models. Consequently, a linear programming solver is all we need. With linear programming solvers we meet the real time constraint that is "less than 3s per solver call". Actually, we obtain 0.25s per solver call. This point is of greatest interest in the real time processes under investigation. Another result of interest is the calculation of a feasible volume's interval. As we'll see in the following, a blend may be infeasible because of its volume. It is then useless to try to modify the recipe while the blend volume is out of the feasible interval.

In this paper we study the blending process with linear blending laws and the components stocked in running tanks. The paper's structure follows. Section 2 presents the Blending Polytopes inherent to the Blending Problem and the Basic Equation which will be used to introduce the updated data in the RTO model. Section 3 analyzes the general case when the heel of a previous blend is used in the production of the new blend and a RTO approach is proposed to solve the Blending Problem. In Section 4 we construct uncertainty sets for various types of uncertainty. These sets determine the robust regions for the blending polytopes in which the new robust RTO (RRTO) method is based. Finally, the price of robustness and reblending is introduced and explained through a real numerical example of the RRTO method in Section 5.

2. Blending Polytopes

Let's denote by n and m the number of components and properties respectively. We will represent a *blend's recipe* (recipe in short) by a vector $u \in R^n$ such that u_j is the percentage of component *j* present in the blend whereas the blend's properties (properties) will be denoted by a vector $y \in R^m$. Then the set of components is described by the $m \times n$ matrix B where $B_{i,j}$ is the i^{th} physical and / or chemical property of the component *j*. It is customary when blending to use the heel of a previous blend to produce a new one. The process starts then with a volume V_0 of a previous blend with properties $b_0 \in R^m$ and continues by adding gradually a volume V of the new blend to obtain a final product's volume $V_{total} = V_0 + V$.

We suppose that a blend's properties are a linear combination (or more precisely, a convex combination) of its component's properties. This is not true in general, but a number of invertible transformations exist such that the transformed properties blend linearly. So, for any $v \in [0, V]$ the blend's properties corresponding to the recipe u are determined by the *basic equation*:

$$y(u) = \alpha_v b_0 + \beta_v B u \qquad (2.1)$$

where, $\alpha_v = \frac{V_0}{V_0+v}, \beta_v = \frac{v}{V_0+v}$.

Having a blend's volume V_0 with properties b_0, equation (2.1) can be used at any time in the process to plan the production of a fixed volume $v \in [0, V]$. We just update b_0 and the matrix B and take $V - v$ as the remaining volume to pour. For this planning horizon, the optimal recipe u is obtained by solving the *Blending Problem* (BP):

$$\min_u c^T u$$
$$\underline{u} \leq u \leq \overline{u} \qquad (2.2)$$
$$\underline{y} \leq y(u) \leq \overline{y} \qquad (2.3)$$
$$1^T u = 1, \; 0 \leq u \leq 1 \qquad (2.4)$$

Here the components' costs are represented by the vector $c \in R^n$. A recipe u is subject to components availability and hydraulic bounds which impose the component's constraints (2.2). Similarly, the properties constraints (2.3) are determined by properties lower $\underline{y}$ and upper ($\overline{y}$) bounds. If $\underline{y_i} = \overline{y}_i$ then we say that property y_i is *regulated*, otherwise it is *controlled*. Additionally, properties' bounds are labeled as *hard* $\left(\underline{y_H}, \overline{y}_H\right)$ or *soft* $\left(\underline{y_S}, \overline{y}_S\right)$. Hard bounds are related to legal, commercial and environmental specifications which must be satisfied whereas soft bounds can be violated but incurring into quality *giveaway*. This means that the blend is better than needed. Finally, (2.4) are the percentage constraints.

We note $H_1 = \{u \in R^n \mid 1^T u = 1\}$ and the simplex $S_1 = \{u \in R^n \mid 1^T u = 1, 0 \leq u \leq 1\}$.

Now, there are 4 distinct polytopes participating in blending process. Geometrically, the intersection of the regions determined by the components constraints and S_1 defines the *recipe's polytope*:

$$U = \left\{u \in S_1 \mid \underline{u} \leq u \leq \overline{u}\right\} \quad (2.5)$$

which represents the set of all recipes that can be produced. On the other hand, the properties constraints define the polyhedron (as it may be unbounded):

$$C_U = \left\{u \in R^n \mid \underline{y} \leq y(u) \leq \overline{y}\right\} \qquad (2.6)$$

Both are defined in the *component's space*.
In *properties space*, the image of U under the basic equation is the *possible blend's polytope*:

$$P = \{y(u) \in R^m \mid u \in U\} \quad (2.7)$$

Finally, the *target polytope* is defined by:

$$C = \left\{y \in R^m \mid \underline{y} \leq y \leq \overline{y}\right\} \quad (2.8)$$

The polytope P depends on the volume v. Let's define $P(\infty) = \{Bu \mid u \in U\}$. Then, $y(u) \in P$ if and only if $y(u) - b_0 = \beta_v(Bu - b_0)$. This shows that $P = P(v)$ is a homothetic transformation of $P(\infty)$ with homothetic center b_0 and homothetic ratio β_v. When v grows continuously from 0 to V, $P(v)$ "moves" continuously from b_0 to $P(V)$. Consequently, if C has a non empty intersection with the cone $cone(b_0, P(\infty))$ defined by the apex b_0 and the base $P(\infty)$ then, there is an interval $v \in [V_{min}, V_{max}]$ such that $P(v) \cap C \neq \emptyset$. In the following we suppose that $cone(b_0, P(\infty)) \cap C \neq \emptyset$. If this were not the case, one has to change the components (and $P(\infty)$) in order to assure feasibility.

Any point in P represents the properties of a blend issued from one or more recipes in U whereas C depicts the target properties to be attained by the blend produced.

We notice that, among the four preceding polytopes, only P and C_U depend on v. We omit the v when it is considered to be fixed.

At this point we have identified the polytopes underlying the BP and we observe that for a fixed volume v, a feasible solution of BP is a recipe $u \in U \cap C_U$ such that $y(u) \in P \cap C$. Equivalently, $BP\ is\ infeasible \iff P \cap C = \emptyset \iff U \cap C_U = \emptyset$.

3. Blending from a previous blend: a RTO method

In any RTO loop, the online optimizer is required to produce a recipe that permits the controller to guide the blending process. In Section 2 we stated that for a fixed volume v to pour, the optimal recipe can be obtained by solving the BP. However, the BP can result to be infeasible because of the uncertainty perturbing the process or for instance, when a previous blend with properties out of specification is used in the production of the new blend. Therefore, a method to compute at any time the best recipe for the planning horizon is required. Later we present this method but first we discuss the BP feasibility when the heel of a previous blend is used.

If the blend is produced from scratch or the volume to pour is too big compared with the heel's volume, that is, if $V_0 = 0$ or $v \to \infty$ then the basic equation reduces to $y(u) = Bu$ and the polytope $P(v)$ does not depend on V_0. However, the fact of using the actual blend's properties b_0 in the basic equation to update the model inside the RTO loop, makes blending from a previous blend the regular case. As explained in the preceding section, we suppose that there is an interval of volumes $[V_{Min}, V_{Max}]$ such that $P(v) \cap C \neq \emptyset, \forall\, v \in [V_{Min}, V_{Max}]$. The BP is feasible only for these volumes.

Having the values V_{Min} and V_{Max} helps to select the appropriate planning horizon at each RTO loop. Choosing a planning horizon by taking a volume $v < V_{Min}$ produces an infeasible BP and makes it necessary to generate an alternative recipe with a possible deterioration of the overall performance of the RTO method. On the other hand, V_{Max} is the maximum volume to blend when looking for a blend within specifications. After V_{Max} has been poured, $P(v)$ moves away from C and the blend's properties deteriorate progressively.

Furthermore, knowing V_{Min} and V_{Max} helps the control system to reduce unnecessary and inefficient interventions. The latter have as purpose to correct the blend to be within specifications but often they are based on a limited view of the problem. In order to compute the interval $[V_{Min}, V_{Max}]$, let's rewrite the basic equation (with $u_0 = \frac{V_0}{v}$):

$$y(u) = \frac{1}{1+u_0}(u_0 b_0 + Bu) \qquad (3.1)$$

The polyhedron C_U becomes $C_U = \left\{u \in R^n \mid (1+u_0)\underline{y} \leq u_0 b_0 + Bu \leq (1+u_0)\overline{y}\right\}$ and the constraints remain linear. Now we present the RTO method:

Blending RTO Method

Let's denote by $C_{HU} = \left\{u \in R^n \mid \underline{y}_H \leq y(u) \leq \overline{y}_H\right\}$ the polyhedron defined by the hard constraints and let $y_T \in C_{HU}$ an "ideal" target blend without quality giveaway provided by the scheduling subsystem.

At any RTO loop, we proceed through the steps I. to VI. as follows:

I. Incorporate the newly available information (V_0, b_0, B, V) in the basic equation. If $V_0 = 0$, solve problems 3.3, 3.6 and 3.7 in this order until having a feasible solution, then STOP. Otherwise, go to step II.

II. Compute V_{Min} and V_{Max} by solving the linear programming problem:

$$\max\,(\min)_{u_0,u}\, u_0,\; u \in U \cap C_U, u_0 \geq 0 \qquad (3.2)$$

If Problem 3.2 is infeasible or if $V \notin [V_{Min}, V_{Max}]$ go to step IV. Otherwise, a blend without quality giveaway exists for any volume $v \in [V_{Min}, V_{Max}]$. Choose a blending horizon $v \in [V_{Min}, V_{Max}]$ and go to step III.

III. Compute the optimal recipe:

$$\min_{\mathrm{u}}\, c^T u\,, u \in U \cap C_U \qquad (3.3)$$

Notice that solving Problem 3.3 with v free and the additional constraint:

$$u_{0Min} \leq u_0 \leq u_{0Max} \quad (3.4)$$

we obtain at the same time the optimal blending horizon $v^* \in [V_{Min}, V_{Max}]$ and the optimal recipe for this volume. This is the best choice if it's not imperative to produce a particular volume. If this is the case STOP. Otherwise go to step IV.

IV. Forget the components costs and focus only on the hard constraints. Compute the interval $[V_{Min}, V_{Max}]$ by solving the problem:

$$\max\,(\min)_{u_0,u}\, u_0,\; u \in U \cap C_{HU}, u_0 \geq 0 \qquad (3.5)$$

If this problem is infeasible or if $V \notin [V_{Min}, V_{Max}]$ go to step VI. Otherwise, go to step V.

V. Look for a recipe $u \in U \cap C_{HU}$ and a volume $v^* \in [V_{Min}, V_{Max}]$ producing the blend with minimal quality giveaway. To do this, solve the following problem with fixed v from a finite set of values, $v \in \{v_1, \dots, v_N\} \subset [V_{Min}, V_{Max}]$:

$$\min_{u,t} \ \|y(u) - y(t)\|_1, u \in U \cap C_{HU}, t \in C_U \qquad (3.6)$$

VI. Find $u \in U$ to produce a volume V of blend with properties as near as possible to the ideal blend y_T:

$$\min_u \ \|y(u) - y_T\|_1, u \in U \qquad (3.7)$$

Following these steps, the RTO method always produces a recipe that guides the control process. In step I, $V_0 = 0$ and solving problems 3.3, 3.6 and 3.7 we obtain the minimal cost recipe, the blend with minimal quality giveaway or the closest blend to y_T . If BP is feasible for some volume $v \in [V_{Min}, V_{Max}]$ then a blend without quality giveaway exists and the method generates the one of minimal cost (step III). Otherwise, the hard constraints become the priority and the method searches for a blend satisfying them while minimizing the blend's quality giveaway (step V). Finally, if there is no blend satisfying the hard constraints, the recipe producing the blend with properties as close as possible to y_T is proposed (step VI).
We finish this section by stressing the possible application of the RTO method to determine the appropriate heel's volume to use in the blend. In our analysis, V_0 is considered as fixed but we can solve the Problem 3.3 with the additional constraint 3.4 with V_0 and V free in order to obtain the cheapest recipe u^* for some $u_0^* \in [u_{0Min}, u_{0Max}]$. Then we can find (by means of the relation $u_0 = \frac{V_0}{v}$) a suitable pair V_0, V. The choice of norm L_1, in problems 3.6 and 3.7 permits to obtain blends violating a minimum number of properties. Moreover, it yields LP problems which can be solved efficiently.

4. Robust RTO

In the previous section we proposed a blending RTO method based on blending polytopes and its evolution with the blended volume. Looking to reduce the model deviations produced by some uncertainties, the method uses blend's and components' properties updates to feedback the model via the basic equation. However, model updating may fail to guarantee even a feasible solution. A main reason of this failure is the implicit assumption that data remains unchanged inside each RTO loop.

For instance, when online blending is used, B values fluctuate with time because components are issued from different process presenting also perturbations. To address this problem, [4] proposed a blending RTO method which updates the model with predictions of the components' properties. Although this method improves the model accuracy, it continues to be non robust as it depends on the quality of the predictions. Moreover, uncertainty in the blending process affects other factors than B values as we will see below.

4.1 BP Uncertainty

In accordance with [6], uncertainty in any RTO system may be of four types:

1. Process uncertainty: components properties, temperature, humidity, etc.
2. Measurement uncertainty.
3. Model uncertainty.
4. Market uncertainty: components availabilities and prices, blends demands, etc.

In this work we consider the uncertainties arising from components measurement and blend's properties measurement (type 2) and the uncertainty caused by imprecise knowledge of the components properties (type 1).

Measurement and components properties uncertainties manifest geometrically in different ways: for the first type, the real recipe (the real percentages of each component used in the blend) and its properties may differ from the nominal ones. So, the real recipe and its properties are located in neighborhoods of the nominal recipe and its properties respectively. For the second type, the real matrix $\tilde{B}$ differs from the nominal matrix B and hence the real polytope $\tilde{C}_U$ is different from the nominal polytope C_U. In both cases, when a nominal feasible recipe $u \in U \cap C_U$ is computed, the real recipe may lie outside the polytope C_U and the real blend's properties may be outside the polytope C. Then the real recipe results to be infeasible.

An intuitive idea to fix our RTO method against measure and B uncertainties follows from the previous geometric information. The idea consists in computing for polytope C_U its convex *robust region* RC_U such that any point in this region resists to B uncertainty and to measurement errors. That is, any point in RC_U is guaranteed to remain inside the real polytope for all possible realizations of B and any measurement error, within reasonable limits.
So, the robust RTO method will consist in replacing polytope C_U by RC_U in the RTO method from Section 3. To develop this idea, first we need to model and measure the uncertainty we would like to be protected against and then to compute the appropriate robust region of C_U. This development follows the Robust Optimization (RO) theory developed by Ben-Tal *et al* (see [1]).

4.2 Robust Regions and Robust RTO

Let u be a nominal recipe and $y(u)$ its properties. To model the components and properties measurement uncertainties, we suppose that the real recipe $\tilde{u}$ lies in the ball $S(u, \delta u)$ of radius δu and center u whereas the real blend's properties $\tilde{y}$ lies in the ball:

$$S(y(u), \delta y) = \{y \in R^m \mid y(u) - \delta y \leq y \leq y(u) + \delta y\}$$

We are given the minimal $\underline{B}$ and maximal $\overline{B}$ values of B. In order to model matrix B uncertainty, we use interval sets. That is, we suppose that each real value $\tilde{B}_{i,j}$ is comprised in the interval $IB_{i,j} = \left[B_{i,j} - \varepsilon_{i,j}^{-}, B_{i,j} + \varepsilon_{i,j}^{+}\right]$ around its nominal value $B_{i,j}$ for some positive values $\varepsilon_{i,j}^{-}, \varepsilon_{i,j}^{+}$.

Here we could use different sets and any norm to model the uncertainty. The level of conservatism (how much we want to be protected against uncertainty) and the problem complexity depend on these selections. Taking interval sets and the max norm, the robust regions obtained are polytopes and the complexity in the model is preserved at the expense of being probably too conservative (we are protected from the worst deviations of all B values and from the biggest measurement errors occurring all at the same time). Now we proceed to construct the robust region of polytope C_U.

Any point $u \in C_U$ is robust regarding B uncertainty iff $\underline{y} \leq \tilde{y}(u) \leq \overline{y}$. That is, iff for any $\tilde{B}$ such that $\tilde{B}_{i,j} \in IB_{i,j}$,

$$\underline{y}_i \leq \alpha_v b_{0,i} + \beta_v \tilde{B}_i u \leq \overline{y}_i, i = 1, \ldots, m \qquad (4.1)$$

holds. Notice that any row $\tilde{B}_i$ can be expressed parametrically as $\tilde{B}_i(z) = B_i + z^{-T} Q_i^- + z^{+T} Q_i^+$, with, $Q_i^- = diag(\varepsilon_{i,1}^-, \ldots, \varepsilon_{i,n}^-), Q_i^+ = diag(\varepsilon_{i,1}^+, \ldots, \varepsilon_{i,n}^+), z^- = \min(0, z)$, $z^+ = \max(0, z)$ for some $z \in R^n$ such that $\|z\|_\infty \leq 1$. Therefore we can deduce that $u \in C_U$ is robust regarding B uncertainty iff

$$\underline{y}_i + \beta_v \varepsilon_i^- u \leq y_i(u) \leq \overline{y}_i - \beta_v \varepsilon_i^+ u, i = 1, \ldots, m \qquad (4.2)$$

In order to derive equation (4.2) let's define for a given u, $f(z) = z^{-T}(Q_i^- u) + z^{+T}(Q_i^+ u)$. We then resolve the following optimization problem:

$$\max_z f(z), \|z\|_\infty \leq 1.$$

The optimal z^0 saturates the constraints. Thus, for any $j = 1, \ldots, n$ one has: either $z^0(j) = 1 = z^{+0}(j)$ or $z^0(j) = -1 = z^{-0}(j)$. The optimal value is $f(z^0) = -\sum_{j, z^0(j)=-1} \varepsilon_{i,j}^- u_j + \sum_{j, z^0(j)=1} \varepsilon_{i,j}^+ u_j$. This is clearly less than or equal to $\|Q_i^+ u\|_1 = \varepsilon_i^+ u$ (because of positivity of ε_i^+ and u) and this gives the right side inequality of (4.2). By resolving $\min_z f(z), \|z\|_\infty \leq 1$ we deduce the left side inequality of (4.2) with a similar reasoning. This is a particular case of a duality result: when resolving $\max_z az, \|z\|_p \leq 1$, the optimum is $\|a\|_q$, where $\|*\|_p$, $\|*\|_q$ are dual norms with $\frac{1}{p} + \frac{1}{q} = 1$.

From now on we denote by $\gamma_i^-(u) = \beta_v \varepsilon_i^- u$ and $\gamma_i^+(u) = \beta_v \varepsilon_i^+ u$.

In addition to B uncertainty, $u \in C_U$ resists also to components' measure uncertainty if and only if any point in $S(u, \delta u)$ satisfies (4.2). That is, iff

$$\underline{y}_i + \gamma_i^-(u) \leq y_i(u + t) \leq \overline{y}_i - \gamma_i^+(u), i = 1, \ldots, m \qquad (4.3)$$

holds for all $t \in R^n$ such that $\|t\|_\infty \leq \delta u$. Then, computing the minimum and maximum on t we get that (4.3) holds iff

$$\underline{y}_i + \gamma_i^-(u) + \delta_i \le y_i(u+t) \le \overline{y}_i - \gamma_i^+(u) - \delta_i, i = 1, \dots, m \quad (4.4)$$

with $\delta_i = \beta_v \delta u \|B_i\|_1$. The reasoning follows the same lines as the deduction of inequalities (4.2).

Equivalently, any recipe $u \in C_U$ is robust regarding B uncertainty and properties' measure uncertainty iff

$$\underline{y}_i + \gamma_i^-(u) \le y_i(u) + Z_i \le \overline{y}_i - \gamma_i^+(u), i = 1, \dots, m \quad (4.5)$$

holds for all $Z \in R^m$ such that $|Z_i| \le \delta y_i$. As previously, computing the minimum and maximum on Z, (4.5) holds if

$$\underline{y}_i + \gamma_i^-(u) + \delta y_i \le y_i(u) \le \overline{y}_i - \gamma_i^+(u) - \delta y_i, i = 1, \dots, m \quad (4.6)$$

Finally, letting $\Delta_i = \max(\delta_i, \delta y_i)$ we obtain the robust region RC_U of polytope C_U:

$$RC_U = \left\{ u \in R^n \mid \underline{y}_i + \gamma_i^-(u) + \Delta_i \le y_i(u) \le \overline{y}_i - \gamma_i^+(u) - \Delta_i, i = 1, \dots, m \right\} \quad (4.7)$$

Any $u \in RC_U$ resists to B uncertainty and to components and properties' measurement uncertainties.

To summarize, let's consider a nominal feasible recipe $u \in U \cap C_U$. If $\tilde{u} \in S(u, \delta u)$ and $\tilde{y} \in S(y(u), \delta y)$ for some $\delta u \in R^+$ and $\delta y \in R^{m+}$ and if $B_{i,j} \in I_{i,j}$, then the recipe $\tilde{u}$ will be feasible in reality: $\tilde{u} \in \tilde{U} \cap \tilde{C}_U$.

The RTO method proposed in Section 3 transforms then in a robust RTO method by a simple substitution of C_U with RC_U. This reduces the impact in implementation as the structure of RTO remains the same. As V and V_0 may be considered as free variables in the RTO method, we can describe the polytope RC_U by using explicitly Equations 4.4 and 4.6 in association with the identity $\beta_v = \frac{1}{1+u_0}$ instead of Equation 4.7 where a Max is involved. This robust RTO method depends completely on the robust region RC_U and to obtain it we only need to determine the values of $\varepsilon^-, \varepsilon^+, \delta u, \delta y$. This study lies upon the fact that this information is available. It's worthwhile to note that while δu and δy are considered as fixed values independent of the RTO loop's length, ε^- and ε^+ depend on it. As fluctuations on B may accumulate over time, the longest the loop's length is, the biggest these fluctuations can be.

Here we limit the analysis to measurement and components properties uncertainties. However, other types of uncertainty manifest geometrically in the same way and thus can be treated identically. For instance, when the uncertainty is due to uncontrolled factors like temperature or humidity, the real blend's properties are located in a ball around the nominal blend's properties. To model the uncertainty in the components prices, we can transform the optimization problem to one with certain objective function and such that uncertainty appears as a constraint ($c^T u \leq \overline{c}$). Then we can construct the uncertainty sets and determine the way it affects the robust regions. We can proceed similarly for the uncertainty in the components availabilities which affects the robust region RU of polytope U.

5. The price of robustness and reblending

In this section we present a case study based on real data to illustrate and compare some key aspects of the RTO method and its robust counterpart. The BP consists in producing a fixed volume $V_{total} = 5000\, m^3$ of blend from 8 components and $V_0 = 2000\, m^3$ of the heel's volume from a previous blend. Each component and the previous blend have 7 properties to be controlled during the process and they are represented by the 7×8 matrix B and vector b_0 respectively. Vectors y_{min} and y_{max} stand for the properties bounds while vector c denotes the components' cost.

	B								b_0	$y_{\min}$	$y_{\max}$
y_1	36.00	36.00	32.00	42.00	16.00	31.00	35.00	46.00	30.00	30.00	46.00
y_2	0.04	0.04	0.03	0.08	0.08	0.14	0.06	0.55	1.66	0.18	1.66
y_3	630.00	620.00	600.00	580.00	620.00	600.00	540.00	450.00	640.00	540.00	640.00
y_4	32.77	32.77	32.77	16.98	16.98	37.72	24.08	8.26	35.19	6.98	35.19
y_5	937.95	937.95	636.62	199.06	199.06	170.47	1381.90	2.80	1381.90	2.02	432.09
y_6	0.80	0.10	0.05	0.04	1.50	2.50	0.05	0.01	1.81	0.00	10.00
y_7	50.00	49.00	50.00	55.00	25.00	39.00	41.00	45.00	40.00	40.00	55.00
c	87.06	87.06	87.02	86.00	83.08	78.05	87.06	117.01			

In order to produce a robust recipe, we assume that components and properties measurement errors are bounded by $\delta y = [0.12, 0.0003,\ 4.5, 0.0826, 0.028, 0.000049, 0.2]$ and $\delta u = 0.01$ respectively. Regarding B uncertainty, we dispose of B^- and B^+ the absolute lower and upper bounds of matrix B. Let's define, $T^- = B - B^-$, $T^+ = B^+ - B$ and $T = B^+ - B^-$. We suppose that there are ε^- and ε^+ such that $B - \varepsilon^- \leq \tilde{B} \leq B + \varepsilon^+$, with $\varepsilon^- = \min(T^-, \theta T)$ and $\varepsilon^+ = \min(T^+, \theta T)$ for some $0 < \theta < 1$. As we stated in Section 4.2, the values of δu and δy are fixed during the process whereas θ is directly related to the RTO loops' length. We take $\theta = 0.01$.

Solving the robust version of problem 3.2 with $V_0 = 2000\, m^3$, we get $[u_{0MinR}, u_{0MaxR}] = [0, 0.0362]$ and the corresponding robust feasible volumes interval: $[V_{MinR}, V_{MaxR}] = [55304, \infty]$. The subscript "$R$" will indicate the result of a robust version of an optimization problem. The corresponding interval for the nominal (non robust) case is $[V_{Min}, V_{Max}] = [34587, \infty]$. This means that we need to produce at least $34587\, m^3$ ($55304\, m^3$) in order to get a (robust) blend within specifications which uses completely V_0. Taking only the hard constraints, we obtain similar intervals.

If we decide to produce $V_{total} = 5000 m^3$ using $V_0 = 2000 m^3$ then we solve the problem 3.7 and we obtain a recipe with cost $90.72\ per\ m^3$ but producing a blend out of specifications. Actually, this blend violates only one property's bound but by more than 80%. Instead of this, we can compute the biggest heel's volume allowing us to produce $5000m^3$ of robust blend. This is the decision taken in practice. From relations $u_{0MaxR} = \frac{V_0}{V}$ and $V_{total} = V_0 + V$ we obtain:

$$V_{0MaxR} = \frac{V_{total} \times u_{0MaxR}}{1 + u_{0MaxR}} = 174.5m^3$$

These $174.5m^3$ are far below the nominal $V_0 = 2000m^3$ forecasted by the scheduling system. The scheduling system uses a linear program to calculate every month the recipes, properties and volumes of reference for a sequence of blends $(u_k^0, y_k^0, V_k^0), k = 1, \dots, K$ (typically $K \in [10,15]$). For the first blends of the sequence, the online process fits well with the forecasted (u_k^0, y_k^0, V_k^0). Nevertheless, after a number of blends, it happens sometimes that the overall blending environment differs significantly from the mean characteristics considered by the scheduling system. In these cases (the present example is one of them), it is very difficult for the online optimizer to take the right decision. Our decision was based on the criterion "minimize reblending".

Next, fixing $V_0 = 174.5m^3$ and $V = 4285.5m^3$ we solve the robust version of problem 3.3 to obtain the optimal robust recipe $u_R^* = [0.1428, 0.0819, 0.0352, 0.1049, 0.2, 0.2, 0.0352, 0.2]$ with cost $c_R^* = 90.72$. Incidentally, this is the same recipe that produces the blend out of specifications!

On the other hand, the optimal nominal recipe

$$u_N^* = [0.1428, 0.0819, 0.0478, 0.169, 0.2, 0.2, 0.0352, 0.1233]$$

has a cost of $c_N^* = 88.35$. Therefore, producing a robust recipe induces a cost increase of 2.68%. However, we observe that if we take $V_0 = 0$ (no reblending), then the robust recipe cost is 87.70 and the nominal recipe cost is 87.04 producing a cost increase of only 0.76%.

From these results we are interested in comparing the price of robustness with the reblending cost (the cost difference between the recipes obtained when the heel's volume is used and when it is not). In order to provide a fair comparison, the price of robustness is obtained by taking $V_0 = 0$ (no reblending involved) and the reblending cost from the nominal recipes (no robustness involved).

To compute the price of robustness we conduct a blending simulation over 36 RTO loops of 2-hours length (10% of the total blend time). In Table 1 we show the average recipe's cost over the 36 periods and the percentage increase in cost (Δc) from the nominal recipe to the robust recipe for different θ values and taking $V_0 = 0$. We used the GLPK solver on an INTEL, 2 CPU, 32bits, 2GHz, 4Gb computer. The execution time for each call was about 0.25s.

We compared this performance (in terms of CPU time) with the classical quadratic model of robustness [1]. This model gives rise to convex programming. We used the CVX toolbox with Matlab (see [8]). The CVX solver took about 15s at each call and this is much above our time constraint (no more than 3s per call).

Then we generate the optimal nominal recipes when $V_0 = 168.44$ (V_{0Max} for $\theta = 0.01$) (with reblending) and $V_0 = 0$ (without reblending) are used in the production of $5000m^3$ of blend. The cost of these recipes are 88.29 and 87.04 respectively. This is normal in general as a "bad" heel is difficult to correct. We observe that for this case, the recipe's cost with reblending is $\frac{(88.29-87.04)}{87.04}$=1.43% greater than the nominal recipe's cost whereas the relative price of robustness is of only 0.35% for a significant value of $\theta = 0.1$.

θ	c_R^*	c_N^*	$\Delta c(\%)$
0.01	87.3851	87.1097	0.32
0.02	87.3542	87.0763	0.32
0.03	87.3847	87.1036	0.32
0.04	87.3632	87.0795	0.32
0.05	87.4058	87.1194	0.33
0.06	87.4134	87.1249	0.33
0.07	87.4232	87.1309	0.33
0.08	87.4258	87.1306	0.34
0.09	87.4297	87.1307	0.34
0.1	87.4077	87.1062	0.35

Table 1: Relative price of robustness for different levels of uncertainty.

6. Conclusions

In this paper a robust real time optimization method for the linear oil blending process has been introduced. The method is based on the RO techniques and it is intended to avoid reblending while minimizing the blend's cost and the quality giveaway. We constructed a set of models for different types of uncertainty arising in the blending process. The simplicity of these models may produce over conservative solutions (blends too expensive) but we showed via an example the convenience of the RO techniques even for these simple models.

A main characteristic of our RRTO method is the integration of the case when the heel of a previous blend has to be incorporated in the new blend. This feature provides meaningful information for the control system, for instance, to determine the appropriate heel's volume to use in the blend or to have an estimate of the volume to pour before getting a blend within specifications.

There are many factors associated with the cost of reblending a previous blend which failed to be within specifications: the tank use, the process time, the inventory costs. Another one is the recipe's cost increase for using the heel's volume of the previous blend. This cost depends (obviously) on the particular blend in which it is used. In this work we compared this cost of reblending with the price of robustness to stress the convenience of the RO techniques.

The results obtained comfort the idea that if reblending or additives cost is expensive enough, then using a more conservative technique like RO may improve the global performance of the blending process. The recipe cost rises by taking the robust recipe in place of the nominal one but reblending and additives expenses cancel. More experiments need to be realized in order to estimate the impact of RTO with a large panel of scenarios.

It appears as a good perspective (kindly suggested by referees) to deal with variable δy and δu especially when longer blending horizons are to be considered. Also, Monte Carlo simulation with variable B could be a way to compare both methods.

Acknowledgements
We thank the referees for their suggestions and remarks.

References

[1] Ben-Tal, A., Ghaoui, L. E., Nemirovski, A., August 2009. Robust Optimization (Princeton Series in Applied Mathematics). Princeton University Press.
[2] Chèbre, M., Crefi, Y., Petit, N., April 2010. Feedback control and optimization for the production of commercial fuels by blending. Journal of Process Control 20 (4), 441-451.
[3] Hendrix, E., July 1996. Finding robust solutions for product design problems. European Journal of Operational Research 92 (1), 28-36.
[4] Singh, A., February 2000. Model-based real-time optimization of automotive gasoline blending operations. Journal of Process Control 10 (1), 43-58.
[5] Wang, W., Li, Z., Zhang, Q., Li, Y., 2007. Online optimization model design of gasoline blending system under parametric uncertainty. In: Proceedings of the 15th Mediterranean Conference on Control & Automation.
[6] Zhang, Y., June 2001. Results analysis for trust constrained real-time optimization. Journal of Process Control 11 (3), 329-341.
[7] Zhang, Y., Monder, D., Fraser Forbes, J., April 2002. Real-time optimization under parametric uncertainty: a probability constrained approach. Journal of Process Control 12 (3), 373-389.
[8] Grant, M., Boyd, S., December 2009. CVX User's Guide.