Robust Grading

Samuel Burer* Veronica Piccialli†

May 8, 2017

Abstract

We propose three strategies by which a professor of a university course can assign final letter grades taking into account the natural uncertainty in students' individual assignment and final numerical grades. The first strategy formalizes a common technique that identifies large gaps in the final numerical grades. For the second and third strategies, we introduce the notion of a *borderline student*, that is, a student who is close to, but below, the breakpoint for the next highest letter grade. Using mixed-integer linear programming and a tailor-made branch-and-bound algorithm, we choose the letter-grade breakpoints to minimize the number of borderline students. In particular, the second strategy treats the uncertainty implicitly and minimizes the number of borderline students, while the third strategy uses a robust-optimization approach to minimize the maximum number of borderline students that could occur based on an explicit uncertainty set. We compare the three strategies on realistic instances and identify overall trends as well as some interesting exceptions. While no strategy appears best in all cases, each can be computed in a reasonable amount of time for a moderately sized course. Moreover, they collectively provide the professor important insight into how uncertainty affects the assignment of final letter grades.

Keywords: Robust optimization; mixed-integer linear programming; branch-and-bound; grading; higher education

1 Introduction

In nearly all educational systems, student performance is measured using *grades*, which assess students' mastery of the topics of instruction. In the American university system specifically, during a typical semester- or quarter-long course, a professor gives numerous assignments, each of which receives a numerical grade, say, on a scale from 0 to 100. At the end of the course, the professor uses a course-specific, student-independent weighting scheme

*Department of Management Sciences, University of Iowa, Iowa City, IA, 52242-1994, USA. Email: samuel-burer@uiowa.edu.

†Dipartimento di Ingegneria Civile e Ingegneria Informatica, Università di Roma Tor Vergata, via del Politecnico 1, 00133, Rome, Italy. Email: veronica.piccialli@uniroma2.it.

to summarize each student's assignment grades into a final numerical grade, which can also be calibrated on a scale from 0 to 100, for example. Finally, the student's final numerical grade is converted to a final letter grade, usually one of A, B, C, D, or F. More finely grained letter grades are also common, e.g., A can be divided into $A+$, A, and $A-$. Traditionally, all students whose numerical final grade is 90 or above receive an A. Of those remaining, 80 or above are assigned a B, 70 or above are assigned a C, etc. However, professors often depart from this traditional conversion from numerical grades to letter grades. For example, a professor might decide that everyone achieving 89 or higher will receive an A.

In general, the process of grading students throughout a course—including assigning the final letter grade—is one of the most important duties of a professor. Not only does grading provide a way to gauge a student's individual performance relative to course goals and to the performance of other students, the final letter grade becomes part of the student's permanent university record, which can affect a student's motivation for continuing at university as well as their education and career prospects later in life. It is thus not surprising that all parties involved (professors, students, universities, and indeed the society at large) have a great interest in accurate and meaningful grading. Rojstaczer and Healy [7] provide a thorough background on the state of modern grading at American universities.

Given that grading is fundamentally a human process, grading is also not surprising that it is subject to considerable uncertainty, leaving professors and students with an important and persistent question: is a student's final letter grade in the course accurate? By the nature of teaching and learning, we can never know with full confidence if a specific grade is accurate. However, as a practical matter, professors and students certainly consider grading uncertainy and its effects. For example: was a particular course assignment too easy or too difficult relative to others; does a particular assignment have a disparate impact on certain subgroups of students; if a course has multiple teaching assistants, do those assistants grade consistently; did a grader make an honest mistake when grading an assignment?

In this paper, we study ways in which a professor for a given course can assign final letter grades taking into consideration the uncertainty that exists in the individual assignment grades. Specifically, we ask what breakpoints the professor should choose for the letter grades (e.g., the traditional breakpoints would be 90, 80, etc.) so that the final letter grades are not overly susceptible to uncertainty in the individual assignment grades.

In Section 2, we discuss two relatively simple techniques that implicitly deal with uncertainty. The first technique involves setting the breakpoints so that the numerical difference between the lowest A grade and highest B grade is maximized—and similarly for B relative to C, C relative to D, etc. We call this *maximizing the gap*. On the other hand, the second technique chooses the breakpoints so that the number of so-called *borderline* students, who

are near—but below—the breakpoints, is minimized. While the first technique requires a simple calculation, the second requires the solution of a mixed-integer linear program [6].

Then, in Section 3, we model uncertainty explicitly by adopting the robust-optimization paradigm [5, 3, 1, 2] using an uncertainty set defined around the (nominal) individual assignment grades. Specifically, our robust approach aims to choose the breakpoints that minimize the worst-case (maximum) number of borderline students. As this problem is a highly nonconvex optimization problem, we develop a tailor-made global branch-and-bound algorithm to optimize the breakpoints.

We note several important aspects of our approach. First, we take the point of view that the professor gives students the "benefit of the doubt," that is, we assume that uncertainty can only increase students' individual assignment grades. Indeed, it is the authors' personal experience that no student will accept consideration of scenarios based on lowering his or her nominal assignment grades. Second, assigning grades is fundamentally a discontinuous process, e.g., a final numerical grade greater than or equal to a breakpoint receives one letter grade, while a grade less than that breakpoint receives another. This discontinuity is reflected throughout the technical details of the paper. Third, in the two techniques mentioned above that involve the optimization of borderline students, we will observe that there are sometimes several optimal solutions. In those situations, we will break ties by using gap maximization as a secondary objective.

Section 4 presents our computational experience with the three strategies described in Sections 2–3. We use random instances that are built to mimic real course data without revealing actual student data, and we employ two realistic types of uncertainty sets. From these instances, we identify an overall trend that the optimal breakpoints provided by the first two strategies (which handle uncertainty implicitly) tend to behave similarly—and often are quite suboptimal—when judged against the objective of the third strategy (which handles uncertainty explicitly). In this sense, taking explicit account of uncertainty noticably affects grading. In addition, we discuss some specific instances that are exceptions to the overall trend.

To the best of our knowledge, there is little to no literature on how professors can assign grades algorithmically taking into account uncertainty. There is, however, a vast literature on testing and assessment in the human-resources and psychological fields; see [9, 8, 4]. Most studies that we encountered focused on the statistical implications of various straightforward selection rules. Thus, our paper appears to be unique in that it utilizes optimization for grade selection as well as the idea of robust (non-statistical) uncertainty.

2 Handling Uncertainty Implicitly

⟨sec:nominal⟩ For m students who have each completed n assignments, the matrix $\bar{G} \in \Re^{m \times n}$ contains the nominal grades $\bar{G}_{ij}$ for each student-assignment pair (i, j). Without loss of generality, each $\bar{G}_{ij}$ is on a common scale; in particular, $\bar{G}_{ij} \in [0, 100]$. Let $w \in \Re^n$ contain the weights of each assignment; w satisfies $w > 0$ and $e^T w = 1$, that is, the sum over all w_j is 1. Then the vector of students' overall grades is $\bar{f} := \bar{G}w$, and each $\bar{f}_i \in [0, 100]$. Finally, a grade in $\{A, B, C, D, F\}$ is assigned to each student based on parameters (or breakpoints) a, b, c, and d: a student gets an A if his/her grade is in the range $[a, 100]$, B if it is in the range $[b, a)$, and so on, with F assigned for any grade below d.

In this paper, we focus on choosing a and b; extensions to more breakpoints is straightforward. In other words, our decision variables are the scalar breakpoints a and b, each of which will also be restricted to suitable bounds $[l_a, u_a]$ and $[l_b, u_b]$, respectively, with $l_a > u_b$. We assume without loss of generality that the students are sorted such that the entries of $\bar{f}$ occur in non-increasing order. Do we actually use this assumption anywhere?

2.1 Maximizing the gap

Suppose the nominal grades $\bar{G}$ and $\bar{f}$ are subject to uncertainty. For fixed a and given student i, the closer $\bar{f}_i$ is to a, the more likely the grade assigned to i will be incorrect due to the uncertainty. Perhaps the student is assigned an A when B is actually the correct grade or *vice versa*. We take the point of view that the decision maker, e.g., the professor, wishes to avoid the situation in which students' final letter grades are underestimated, e.g., when $a = 90$, $\bar{f}_i = 89.9$, a B is assigned, and yet due to uncertainty in $\bar{G}$ and $\bar{f}$, the student's true letter grade should be A. In essence, the assigned B accidentally penalizes the student. In addition, it could lead to dissatisfaction and a complaint from that student, which the professor would like to avoid.

When choosing the value of a, an obvious strategy—without explicitly taking into account grade uncertainty—is to choose the value that maximizes the gap between a and the first student who receives a B, that is, to choose a by solving

$$\begin{aligned} \text{maxgap}_a := \max_a \quad & \text{gap}(a) := \min_{i=1}^{m} \{a - \bar{f}_i : \bar{f}_i < a\} \\ \text{s.t.} \quad & l_a \le a \le u_a. \end{aligned} \tag{1}$$ eq:pmgap

Let a_{gap} denote the optimal solution of (1); we call this the *maxgap solution with respect to* a. (If there is more than one optimal solution, we take the largest one; see the next paragraph.) We can also similarly define maxgap_b and b_{gap}. Clearly, since $\bar{f}$ is fixed in this context, i.e.,

we are not explicitly taking into account uncertainty, and since $l_a > u_b$ by assumption, the calculation of $(a_{\text{gap}}, b_{\text{gap}})$ is separable. Considered together, the overall maxgap value is the minimum of the two separate values, i.e., we define

$$\text{maxgap}_{(a,b)} := \min\{\text{maxgap}_a, \text{maxgap}_b\}.$$

It is in fact straightforward to calculate a_{gap}. It must happen that $\text{gap}(a)$ is maximized at some $\bar{f}_i$ or at u_a. If not, it would be possible to increase $\text{gap}(a)$ by simply increasing a and stopping at the next larger $\bar{f}_i$ or at u_a to maintain feasibility. More formally, for any interval $[l, u]$, let $\bar{F}(l, u) = \{\bar{f}_i : f_i \in [l, u]\}$ denote the set of grades belonging to the interval $[l, u]$. Then:

Proposition 1. *The optimal pair* $(a_{\text{gap}}, b_{\text{gap}})$ *satisfies* $a_{\text{gap}} \in \bar{F}(l_a, u_a) \cup \{u_a\}$ *and* $b_{\text{gap}} \in \bar{F}(l_u, l_b) \cup \{u_b\}$.

This proposition implies that $(a_{\text{gap}}, b_{\text{gap}})$ can be calculated by simple enumeration, and this algorithm will be referred to as *MaxGap* in Section 4.

2.2 Borderlines

In contrast to the maxgap solution a_{gap}, the professor might choose a in order to minimize the total number of students who are below, but very close to, a. We call these *borderlines*. Borderline students are those who, taking into account grade uncertainty, might be assigned the wrong final grade or might be dissatisfied with their final grades. The borderline concept depends on a parameter $\epsilon > 0$ which defines the meaning of "close." For example, suppose we have six students with nominal grades $\bar{f}_i$ as follows: 90.3, 89.6, 89.1, 89.1, 89.1, 89.1, 88.9. Taking $l_a = 88$ and $u_a = 90$, we have $a_{\text{gap}} = 89.6$ so that the maximum gap with respect to a is 0.5. However, this choice for a yields four borderlines if "close" means being within $\epsilon = 0.5$ of a. On the other hand, choosing $a = 90$ still has a good gap of 0.4, but only one student is borderline.

More precisely, given $a \in [l_a, u_a]$, we say that student i is *borderline with respect to* a if

$$\bar{f}_i < a \leq f_i + \epsilon \quad \iff \quad \bar{f}_i \in [a - \epsilon, a) \tag{2}$$ borderline

where $\epsilon > 0$ is the *borderline tolerance* chosen by the decision maker such that $u_b + \epsilon < l_a$. In words, the student's nominal grade $\bar{f}_i$ is less than a, but adding ϵ to $\bar{f}_i$ results in a grade that meets or exceeds a. We would like to choose a so that the total number of students

who are borderline with respect to a is minimized. Precisely, defining

$$L(a, \bar{f}_i) := \begin{cases} 1 & \text{if } a \text{ and } \bar{f}_i \text{ satisfy (2)} \\ 0 & \text{otherwise,} \end{cases}$$

we would like to minimize the total number of borderlines $\sum_{i=1}^{m} L(a, \bar{f}_i)$ over $a \in [l_a, u_a]$. In a similar manner, for the choice of the second breakpoint b, we wish to minimize $\sum_{i=1}^{m} L(b, \bar{f}_i)$ over $b \in [l_b, u_b]$. Due to the assumption $u_b + \epsilon < l_a$, we may join these separate problems into a single minimization:

$$\min_{(a,b)\in X} \sum_{i=1}^{m} \left[L(a, \bar{f}_i) + L(b, \bar{f}_i) \right]$$

where $X := [l_a, u_a] \times [l_b, u_b]$ is the rectangle of feasible breakpoints.

Let us further formalize the definition of $L(a, \bar{f}_i)$. For variable scalars $a, \varphi \in [0, 100]$, define the *borderline indicator function*

$$L(a, \varphi) := \begin{cases} 0 & \text{if } \varphi < a - \epsilon \\ 1 & \text{if } \varphi \in [a - \epsilon, a) \\ 0 & \text{if } \varphi > a; \end{cases}$$

see Figure 1. In words, $L(a, \varphi)$ is 1 if and only if φ is borderline with respect to a. Note that $L(a, \varphi)$ is a discontinuous, piece-wise linear function even when a is fixed, and because of these discontinuities, it is challenging to model $L(a, \varphi)$ using mixed-integer linear programming.

Indeed, in [10, 11] it is shown that the epigraph of a lower semicontinuous function can be represented by a mixed integer programming problem for the purpose of minimization, but we will need both to maximize and minimize the number of pseudo-borderlines students (see Section 3), so we cannot apply this approach directly.

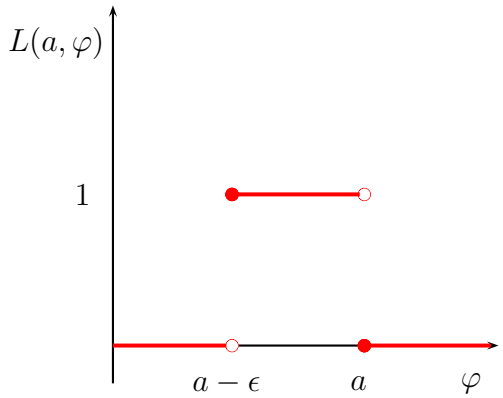

Figure 1: Borderline indicator function

ig:borderline⟩

Because the borderline function $L(a, \varphi)$ cannot be captured for our purposes with mixed-integer linear programming, we suggest a modified approach based on approximating the borderline concept as we discuss next.

2.3 Pseudo-borderlines

⟨ssec:pseudo⟩

For a given *pseudo-borderline tolerance* $0 < \delta \ll \epsilon$, we define a continuous approximation of $L(a, \varphi)$ called the *pseudo-borderline indicator function*:

$$P(a,\varphi) := \begin{cases} 0 & \varphi \le a - \epsilon - \delta \\ \delta^{-1}(\varphi - a + \epsilon + \delta) & \varphi \in [a - \epsilon - \delta, a - \epsilon] \\ 1 & \varphi \in [a - \epsilon, a - \delta] \\ -\delta^{-1}(\varphi - a) & \varphi \in [a - \delta, a] \\ 0 & \varphi \ge a; \end{cases}$$

see Figure 2. Note that $P(a, \varphi)$ is continuous, and it approximates $L(a, \varphi)$ well for small δ. Hence the total number of borderlines will be approximated well by the total number of pseudo-borderlines, i.e.,

$$\sum_{i=1}^{m} L(a, \bar{f}_i) \approx \sum_{i=1}^{m} P(a, \bar{f}_i).$$

Note that we could define other, closely related variants of $P(a, \varphi)$ for approximating $L(a, \varphi)$, e.g., one which minorizes L or one which majorizes L.

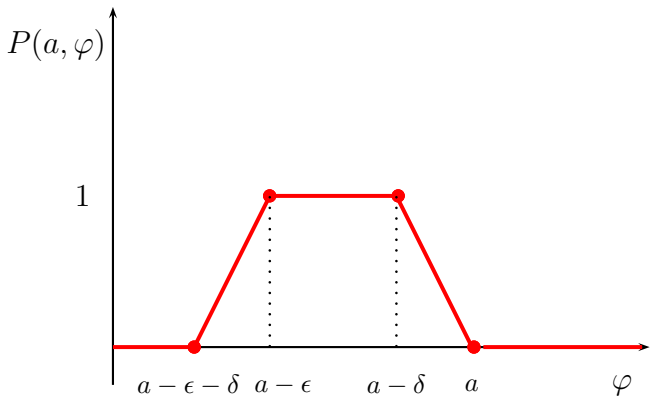

Figure 2: Pseudo-borderline Indicator function

udoborderline⟩

As motivated in the prior subsection, we choose $P(a, \varphi)$ instead of $L(a, \varphi)$ because it can be represented using mixed-integer linear programming. Before describing this representation, we note that $P(a, \varphi)$ can itself be written as the difference of two related functions.

Define

$$Q(a,\varphi) = \begin{cases} 0 & \text{if } \varphi \le a-\delta \\ \delta^{-1}(\varphi - a + \delta) & \text{if } \varphi \in [a-\delta, a] \\ 1 & \text{if } \varphi \ge a. \end{cases}$$

In words, $Q(a,\varphi)$ essentially indicates whether φ exceeds a. Since φ is borderline with respect to a if and only if it exceeds $a-\epsilon$ but does not exceed a, we have $P(a,\varphi) = Q(a,\varphi+\epsilon) - Q(a,\varphi)$, and hence we would like to minimize

$$\sum_{i=1}^{m} L(a,\bar{f}_i) \approx \sum_{i=1}^{m} P(a,\bar{f}_i) = \sum_{i=1}^{m} \left[Q(a,\bar{f}_i+\epsilon) - Q(a,\bar{f}_i)\right].$$

From a modeling perspective, this representation is convenient because we can model the number of pseudo-borderlines using $2m$ building blocks each of the form $Q(a,\varphi)$.

In particular, we will model the function $Q(a,\varphi)$ as a relationship in 3-dimensional space. Consider variables

$$a \in [l_a, u_a], \qquad \varphi \in [l_\varphi, u_\varphi], \qquad \alpha \in [0,1],$$

where the general bounds $[l_a, u_a] \subseteq [0,100]$ and $[l_\varphi, u_\varphi] \subseteq [0,100]$ will be convenient for modeling purposes throughout the remainder of the paper. Then $Q(a,\varphi)$ can be expressed as the union of three 2-dimensional polyhedra in the variable triples (a,φ,α), each of which represents one piece of the piece-wise linear relationship defined by $Q(a,\varphi)$:

$$\mathcal{Q}^1(l_a,u_a,l_\varphi,u_\varphi) := \left\{ (a,\varphi,\alpha) : \begin{array}{l} a \in [l_a,u_a],\ \varphi \in [l_\varphi,u_\varphi] \\ \varphi \le a-\delta,\ \alpha = 0 \end{array} \right\},$$

$$\mathcal{Q}^2(l_a,u_a,l_\varphi,u_\varphi) := \left\{ (a,\varphi,\alpha) : \begin{array}{l} a \in [l_a,u_a],\ \varphi \in [l_\varphi,u_\varphi] \\ a-\delta \le \varphi \le a,\ \alpha = \delta^{-1}(\varphi - a + \delta) \end{array} \right\},$$

$$\mathcal{Q}^3(l_a,u_a,l_\varphi,u_\varphi) := \left\{ (a,\varphi,\alpha) : \begin{array}{l} a \in [l_a,u_a],\ \varphi \in [l_\varphi,u_\varphi] \\ a \le \varphi,\ \alpha = 1 \end{array} \right\}.$$

With these definitions, the problem of minimizing the number of pseudo-borderline students with respect to a becomes

$$\begin{array}{lll} \min\limits_{a} & \displaystyle\sum_{i=1}^{m} P(a,\bar{f}_i) = \sum_{i=1}^{m} \left[Q(a,\bar{f}_i+\epsilon) - Q(a,\bar{f}_i)\right] = \sum_{i=1}^{m} [\alpha_i^\epsilon - \alpha_i] & \\ \text{s.t.} & (a,\bar{f}_i,\alpha_i) \in \bigcup_{k=1}^{3} \mathcal{Q}^k(l_a,u_a,\bar{f}_i,\bar{f}_i) & i = 1,\dots,m \\ & (a,\bar{f}_i+\epsilon,\alpha_i^\epsilon) \in \bigcup_{k=1}^{3} \mathcal{Q}^k(l_a,u_a,\bar{f}_i+\epsilon,\bar{f}_i+\epsilon) & i = 1,\dots,m \\ & l_a \le a \le u_a. & \end{array} \quad (3)$$

eq:nomprob

Note that the generic interval $[l_\varphi, u_\varphi]$ is a singleton, either $[\bar{f}_i, \bar{f}_i]$ or $[\bar{f}_i + \epsilon, \bar{f}_i + \epsilon]$, in each appearance of $\mathcal{Q}^k(l_a, u_a, l_\varphi, u_\varphi)$ above. This forces each corresponding φ to equal either $\bar{f}_i$ or $\bar{f}_i + \epsilon$, which is consistent with the fixed nominal data $\bar{f}$. In Section 3, we will allow $[l_\varphi, u_\varphi]$ to have positive width, which will be consistent with the case that there is explicit uncertainty in the grades.

In problem (3), the constraints $(a, \varphi, \alpha) \in \bigcup_{k=1}^{3} \mathcal{Q}^k(l_a, u_a, l_\varphi, u_\varphi)$ can be modeled using a standard mixed-integer approach. For each $k = 1, 2, 3$, we introduce a quadruple $(a^k, \varphi^k, \alpha^k, t^k)$, which belongs to the homogenization of the polyhedron $\mathcal{Q}^k$, where t^k is the binary homogenization variable:

$$
\begin{aligned}
t^1 \in \{0,1\}, \quad & l_a t^1 \le a^1 \le u_a t^1, \quad l_\varphi t^1 \le \varphi^1 \le u_\varphi t^1 \\
& \varphi^1 \le a^1 - \delta t^1, \quad \alpha^1 = 0, \\
t^2 \in \{0,1\}, \quad & l_a t^2 \le a^2 \le u_a t^2, \quad l_\varphi t^2 \le \varphi^2 \le u_\varphi t^2 \\
& \alpha^2 = \delta^{-1}(\varphi^2 - a^2 + \delta t^2), \quad a^2 - \delta t^2 \le \varphi^2 \le a^2, \\
t^3 \in \{0,1\}, \quad & l_a t^3 \le a^3 \le u_a t^3 \quad l_\varphi t^3 \le \varphi^3 \le u_\varphi t^3 \\
& a^3 \le \varphi^3, \quad \alpha^3 = t^3.
\end{aligned}
$$

We finally enforce $(a, \varphi, \alpha, 1) = \sum_{k=1}^{3} (a^k, \varphi^k, \alpha^k, t^k)$ to ensure that (a, φ, α) is contained in exactly one of $\mathcal{Q}^1$, $\mathcal{Q}^2$, or $\mathcal{Q}^3$. The resulting formulation of (3) contains $14m + 1$ continuous variables and $6m$ binary variables. Moreover, there are $8m$ linear equality constraints and $30m$ linear inequality constraints. A similar, separate problem can be defined for determining the optimal breakpoint b.

2.4 An algorithm to optimize pseudo-borderlines (and maxgap)

⟨sec:basic⟩

As discussed, problem (3) can be solved directly with a mixed-integer-programming solver. However, due to the fact that its objective function $\sum_{i=1}^{m} P(a, \bar{f}_i)$ is integer valued for many values of a, it is likely that (3) has multiple optimal solutions. Indeed, we have observed this in practice. Hence, we have implemented a simple divide-and-conquer algorithm to maximize $\mathrm{gap}(a)$ as a secondary objective.

Specifically, we first solve (3) to calculate the minimum value P^* of pseudo-borderlines over the entire interval $l_a \le a \le u_a$. We then sub-divide the interval and calculate the minimum value p^* on the sub-intervals as well. For each sub-interval, if $p^* > P^*$, then we discard that sub-interval since it clearly does not contain any optimal values of a. Otherwise, it must hold that $p^* = P^*$, and we have obtained an optimal a^* in that sub-interval; we then divide that sub-interval even further. We continue this process of creating and discarding

sub-intervals until each remaining sub-interval is below a certain user-specified width $\omega > 0$ and contains a verified optimal a^*. We then return the optimal a^* among all optimal solutions having the maximum value of gap(a).

A similar algorithm is used to calculate b. In Section 4, we will refer to this overall procedure to calculate a and b with minimum psuedo-borderlines (and good maxgap values) as the *Basic* algorithm.

3 Handling Uncertainty Explicitly

ec:robustcase⟩

3.1 Minimizing the maximum borderlines

Recall that the matrix $\bar{G}$ contains the nominal grades. We assume that, in reality, the grades matrix is subject to uncertainty, and we let $G \in \mathcal{U} \subseteq \Re^{m \times n}$ represent this uncertain matrix, where $\mathcal{U}$ is a user-defined, compact uncertainty set. We assume that $\mathcal{U}$ can be represented as a mixed-integer feasible set and that $G \in \mathcal{U}$ satisfies $G \geq \bar{G}$, that is, the uncertainty can only increase the grades compared to their nominal values, which is consistent with a cautious professor who does not wish to consider the possibility of decreasing student grades from their nominal values. Moreover, since $\mathcal{U}$ is compact, we may pre-calculate, for each student i, lower and upper bounds for f_i:

$$l_{f_i} := \min_{G \in \mathcal{U}, f = Gw} f_i \qquad \text{and} \qquad u_{f_i} := \max_{G \in \mathcal{U}, f = Gw} f_i. \tag{4}$$ equ:bounds

These bounds will prove useful in the formulation (6) below and in a pre-processing stage for instances discussed in Section 4.

To account for the uncertainty $G \in \mathcal{U}$, we would like to determine the values of the break points (a, b) that minimize the maximum (worst-case) number of pseudo-borderline students:

$$\min_{(a,b) \in X} \max_{G \in \mathcal{U}, f = Gw} \sum_{i=1}^{m} \left[P(a, f_i) + P(b, f_i) \right]. \tag{5}$$ robprob-tot

Note that both G and f are variables in (5). Using the development of Section 2.3 and the

similarities with (3), we can write (5) explicitly as

$$\begin{array}{lll} \min\limits_{(a,b)\in X} & \max_{f=Gw,G\in\mathcal{U}} & \sum\limits_{i=1}^{m}[\alpha_i-\alpha_i^\epsilon]+\sum\limits_{i=1}^{m}[\beta_i-\beta_i^\epsilon] \\ & \text{s.t.} & (a,f_i,\alpha_i)\in\bigcup_{i=1}^{3}\mathcal{Q}^k(l_a,u_a,l_{f_i},u_{f_i}) \qquad i=1,\ldots,m \\ & & (b,f_i,\beta_i)\in\bigcup_{i=1}^{3}\mathcal{Q}^k(l_b,u_b,l_{f_i},u_{f_i}) \qquad i=1,\ldots,m \\ & & (a,f_i+\epsilon,\alpha_i^\epsilon)\in\bigcup_{i=1}^{3}\mathcal{Q}^k(l_a,u_a,l_{f_i}+\epsilon,u_{f_i}+\epsilon) \quad i=1,\ldots,m \\ & & (b,f_i+\epsilon,\beta_i^\epsilon)\in\bigcup_{i=1}^{3}\mathcal{Q}^k(l_b,u_b,l_{f_i}+\epsilon,u_{f_i}+\epsilon) \quad i=1,\ldots,m. \end{array} \tag{6}$$ eq:robprob

Note that the bounds l_{f_i} and u_{f_i} are used in the representation of the $\mathcal{Q}^k$ polyhedron to provide the required bounds on the φ variable; see also the discussion in Section 2.3.

We note that problem (6) is not separable, i.e., it is coupled between a and b via the uncertainty set $\mathcal{U}$ and the variables G and f, which are common to all constraints in (6).

3.2 A spatial branch-and-bound algorithm

We solve (6) using a tailored spatial branch-and-bound method based on subdividing the rectangular domain $X=[l_a,u_a]\times[u_a,u_b]$ into smaller and smaller rectangles $\hat{X}$. The algorithm also utilizes subroutines to compute upper and lower bounds for the min-max objective at a given node in the branch-and-bound tree, that is, for problem (6) restricted to $\hat{X}$.

For a given sub-rectangle $\hat{X}$, we compute an upper bound on the optimal value of (6) restricted to $\hat{X}$ by simply fixing some $(\hat{a},\hat{b})\in\hat{X}$—typically the midpoint—and solving the inner maximization problem. The smallest of all these upper bounds is the global upper bound (GUB), which is maintained throughout the course of the algorithm and equals the optimal value upon termination.

A byproduct of the upper-bound computation is an optimal solution $\hat{f}$, which yields the worst-case number of pseudo-borderlines for $(\hat{a},\hat{b})$. We may consider $\hat{f}$ to be a reasonable candidate for the worst-case f over all (a,b). Morever, $\hat{f}$ can be used to compute a lower bound on the optimal value of (6) restricted to $\hat{X}$ by solving $\min_{(a,b)\in\hat{X}}[P(a,\hat{f})+P(b,\hat{f})]$. Indeed, let $\mathcal{B}$ denote a collection (or "bunch") of such candidate $\hat{f}$ collected from various smaller rectangles throughout the course of the algorithm. Then we can calculate an improved lower bound by solving

$$\max_{\hat{f}\in\mathcal{B}}\ \min_{(a,b)\in\hat{X}}[P(a,\hat{f})+P(b,\hat{f})], \tag{7}$$ robprob-lb

which is simply a compact way to state that we pick the maximum lower bound from a finite list of lower bounds, each tailored to $\hat{X}$.

Then our overall strategy to compute strong lower bounds during the course of the algorithm is to update $\mathcal{B}$ continually by adding a new $\hat{f}$ at every upper bound computation. We also keep track of which $\hat{f} \in \mathcal{B}$ are particularly helpful, i.e., those that tend to achieve the best lower bounds often for different sub-rectangles $\hat{X}$. We keep the helpful $\hat{f}$ and discard others in order to keep size of $\mathcal{B}$ small. We also calculate (7) in order from most-helpful $\hat{f}$ to least-helpful in order to increase the chance of fathoming $\hat{X}$ as early as possible via a strong lower bound which exceeds the current GUB.

We process the nodes in the branch-and-bound tree using the standard best-bound rule. In addition, our branching strategy is the standard one from continuous global optimization, i.e., each interval is halved, and the algorithm stops when the diagonal length of all remaining rectangles is below a certain user-specified *width threshold* $\omega > 0$.

Finally, during the course of the algorithm, if we detect a sub-rectangle $\hat{X}$ in which its lower bound equals the current GUB, then we know that every $(a, b) \in \hat{X}$ achieves the GUB—and hence there is no reason to explore $\hat{X}$ further. (This situation occurs quite often in our experiments.) On the other hand, every pair (a, b) in $\hat{X}$ is a candidate optimal solution for (6). We save the pair which maximizes $\text{maxgap}_{(a,b)}$ on $\hat{X}$. This is similar to the tie-breaking using maxgap discussed in Section 2.4.

The overall algorithm is referred to as *Robust* in the following section.

4 Computational Results

c:compresults⟩ To test the algorithms discussed in Sections 2 and 3, we experiment with realistic data from an actual course, but we do so in a manner that protects individual student privacy. Specifically, beginning with a real-world grades matrix from one of the first author's courses, which had $m = 41$ students and $n = 5$ assignments, we calculated the empirical distribution for each of the assignments. We then simulated students in the course by sampling m sets of grades for the assignments independently, which provided a single nominal $\bar{G} \in \mathbb{R}^{m \times n}$. We then repeated the same experiment to generate a total of 50 nominal matrices $\bar{G}$, which became the basis of our numerical experiments below. In particular, the original course data was not used to test the algorithms directly; it was simply the basis for randomly generating the test data. While clearly our approach only simulates the original course approximately—for example, we do not take into account correlations between the individual assignments—it is straightforward, is inspired by a real-world classroom, and protects student privacy.

We consider uncertainty sets $\mathcal{U} \ni G$ that bound the norm of the nonnegative difference matrix $G - \bar{G} \geq 0$. First, as a ground set, we allow each grade entry G_{ij} to increase up to 5 points above $\bar{G}_{ij}$ without exceeding 100, that is, $G_{ij} \leq \min\{\bar{G}_{ij} + 5, 100\}$. Precisely, let E be

the all-ones matrix, and define the matrix $\hat{G} := \min\{\bar{G} + 5E, 100E\}$ component-wise. Then the inequalities $\bar{G} \leq G \leq \hat{G}$ define our ground set. Next, for a discrete choice $k \in \{0, 1\}$ and a continuous choice $c \in [0, 1]$, we define the generic uncertainty set

$$\mathcal{U}(k, c) := \{\bar{G} \leq G \leq \hat{G} : \|G - \bar{G}\|_k \leq c\|\hat{G} - \bar{G}\|_k\}.$$

Here, for a matrix $M \geq 0$, the 1-norm is defined $\|M\|_1 := \sum_{ij} M_{ij}$, and the 0-norm counts the number of positive entries, that is, $\|M\|_0 := |\{(i, j) : M_{ij} > 0\}|$. In words, $G \in \mathcal{U}(k, c)$ limits G to be some fraction c of the full perturbation $\hat{G}$ from G, where c is measured relative to the k-norm. In particular, $\mathcal{U}(0, c)$ limits the fraction of entries of G that are different from their counterparts in $\bar{G}$. Note that $\mathcal{U}(1, c)$ has a polyhedral representation:

$$\mathcal{U}(1, c) := \left\{\bar{G} \leq G \leq \hat{G} : \sum_{ij}(G_{ij} - \bar{G}_{ij}) \leq c\|\hat{G} - \bar{G}\|_1\right\}.$$

Also, $\mathcal{U}(0, c)$ has the following mixed-integer formulation:

$$\mathcal{U}(0, c) := \left\{\bar{G} \leq G \leq \hat{G} : \begin{array}{l} G - \bar{G} \leq Y \circ (\hat{G} - \bar{G}), Y_{ij} \in \{0, 1\} \\ \sum_{ij} Y_{ij} \leq c\|\hat{G} - \bar{G}\|_0 \end{array}\right\},$$

where $\circ$ indicates component-wise multiplication.

Given the nominal grades $\bar{G}$ and $\bar{f} := \bar{G}w$, as well as a specific uncertainty set $\mathcal{U}$, we preprocess the data in order to detect any students that—for all feasible a and for all grade realizations $G \in \mathcal{U}$—can never be pseudo-borderline with respect to a. Said differently, we determine all i such that $P(a, f_i) = 0$ for all a and all G, where $f := Gw$. For example, this might be a student whose nominal $\bar{f}_i$ equals $u_b + 1$ and yet, even in the most extreme scenario, her grade cannot reach $l_a - \epsilon - \delta$, i.e., she cannot be pseudo-borderline with respect to a. Then, since $P(a, f_i)$ is identically zero, we can simplify (5) and its implementation (6). To determine such students, for each i, we first compute the minimum and maximum values of f_i over the uncertainty set according to (4). We then deduce $P(a, f_i) = 0$ for all a and G when either of the following conditions is met: $u_{f_i} < l_a - \epsilon - \delta$; or $l_{f_i} > u_a$. In the first case, student i is guaranteed a B letter grade, and in the second, she is guaranteed an A letter grade. We do a similar preprocessing with respect to b. Note also that the constants l_{f_i} and u_{f_i} are required for the full specification of (6).

Sections 2 and 3 have introduced three methods for choosing the breakpoints (a, b): (i) maximizing the gap with respect to the nominal $\bar{f}$, a method we call *MaxGap*; (ii) minimizing the number of pseudo-borderlines with respect to the nominal $\bar{f}$, called *Basic*;

and (iii) minimizing the worst-case (maximum) number of pseudo-borderlines for uncertain $G \in \mathcal{U}$, called *Robust*. We run MaxGap, Basic, and Robust on the 50 instances described above separately for eight parameter combinations: $(\epsilon, k, c) \in \{0.5, 1.0\} \times \{0, 1\} \times \{0.1, 0.2\}$. Recall that $\epsilon > 0$ is the borderline tolerance chosen by the decision maker.

Furthermore, for all instances, we set the lower and upper bounds for b to be $(l_b, u_b) := (78, 80)$; the bounds for a are $(l_a, u_a) := (88, 90)$. We set the pseudo-borderline tolerance (discussed in Section 2.3) to $\delta = 0.001$ for all instances, and the width tolerance (see Section 3) equals $\omega = 0.09$ always. In addition, we use a tolerance 0.001 for judging whether two pseudo-borderline function values are equal within the branch-and-bound algorithm, i.e., for the purposes of fathoming.

We coded our branch-and-bound algorithm in Matlab R2016b with calls to Gurobi 6.5.1 for solving the mixed-integer subproblems. All runs were executed on a distributed cluster at the University of Iowa, but each run used only one processor/core running on CentOS 6.4 at 2.6GHz and with access to 64GB of shared RAM. The run times for MaxGap and Basic were small, while the times for Robust broke into two distinct groups based on the norm parameter $k \in \{0, 1\}$ for the uncertainty set. For $k = 1$, the run times were log-normally distributed with parameters $(\mu, \sigma) = (2.70, 0.26)$ (base 10), and for $k = 0$, the run times were also log-normally distributed but with parameters $(\mu, \sigma) = (3.73, 0.42)$. This means, for example, that a typical run of Robust finished within no more than $10^{3.73} \approx 5370$ seconds.

4.1 Overall trends

Since each method optimizes its own objective function, we compare the methods by examining how each method's solution performs when plugged into the others' objective functions. Specifically, we examine the sub-optimality (either relative or absolute) of one solution in another's objective function. Figures 3 and 4 are bubble plots depicting these comparisons; in each plot, the size of a bubble is proportional to the number of instances at those coordinates. For example, for $\epsilon = 0.5$, the left-most bubble plot of Figure 3 depicts the sub-optimality (in percentage terms) of the solutions (a, b) provided by MaxGap and Basic when those solutions are inserted into the objective function of Robust. Note that results in this bubble plot are aggregated over the four combinations $(k, c) \in \{0, 1\} \times \{0.1, 0.2\}$, which in particular means that each bubble plot corresponds to $200 = 4 \cdot 50$ instances.

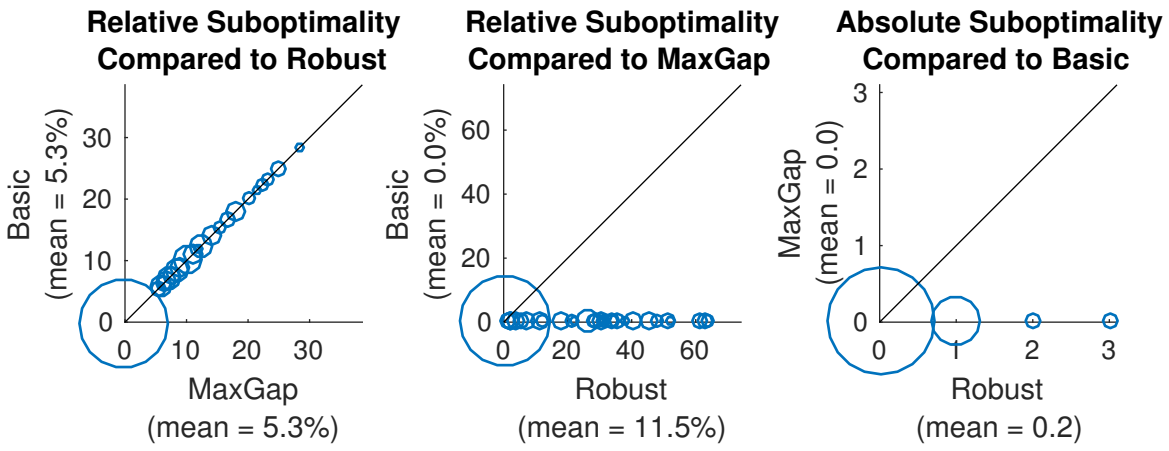

Figure 3: Bubble plots for $\epsilon = 0.5$ and $(k, c) \in \{0, 1\} \times \{0.1, 0.2\}$

⟨fig:bubble05⟩

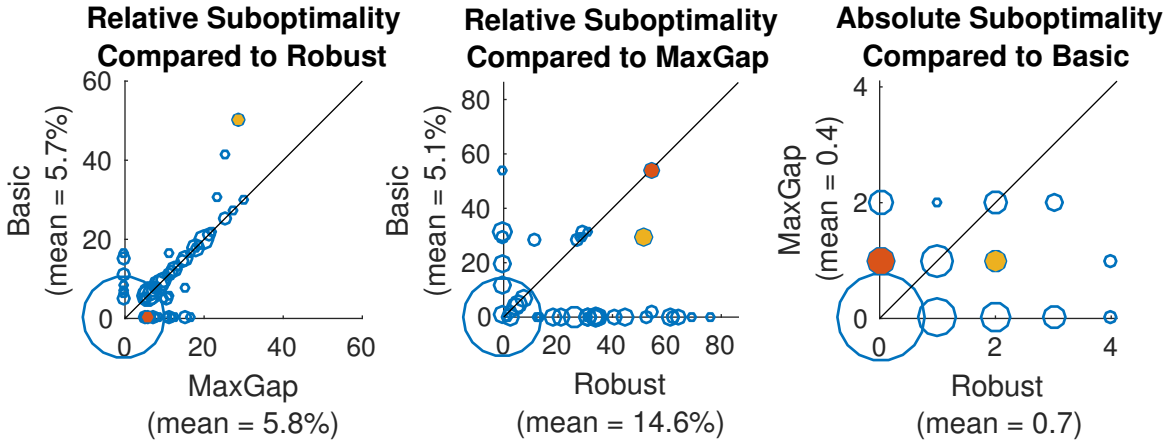

Figure 4: Bubble plots for $\epsilon = 1$ and $(k, c) \in \{0, 1\} \times \{0.1, 0.2\}$

⟨fig:bubble1⟩

In the left-most bubble plot of Figure 3, we see that MaxGap and Basic: are typically 5.3% sub-optimal with respect to Robust; perform the same as each other since all bubbles are positioned on the "$y = x$" line; and are most typically optimal for Robust as indicated by the largest bubble at the origin. The left-most bubble plot in Figure 4 shows the same comparison except for $\epsilon = 1.0$. Even with this larger borderline tolerance, we see approximately the same average sub-optimality of Basic and MaxGap with respect to Robust, but there is now some variation in the results. In particular, Basic and MaxGap do not always perform the same, and they are not as often optimal for Robust. This different behavior between Figures 3 and 4 is not surprising since one would expect the higher value of ϵ to result in more variation between the methods.

The center plots of Figures 3 and 4 compare the relative sub-optimality of Basic and Robust with respect to MaxGap. Consistent with the previous paragraph, Basic performs the same as MaxGap when $\epsilon = 0.5$, and often the same as MaxGap when $\epsilon = 1.0$. On the other hand, Robust is on average 11.5% sub-optimal with respect to MaxGap when $\epsilon = 0.5$ and 14.6% sub-optimal when $\epsilon = 1.0$. Nevertheless, there are many individual cases when Robust is optimal for MaxGap, too.

Finally, the right-most bubble plots of Figures 3 and 4 compare the performance of MaxGap and Robust with respect to Basic. In contrast to the first two pairs of bubble plots, we present absolute sub-optimality because the objective value of Basic is typically a small whole number, e.g., between 0 and 4, and we believe an absolute measure is easier to interpret in this case. Again consistent with the other bubble plots, we see that MaxGap and Basic behave similarly, whereas Robust and Basic are more different.

Overall, it emerges from Figures 3 and 4 that Basic and MaxGap, which are both calculated quickly using only the nominal data, behave similarly in both the nominal and robust situations. Moreover, we can see that they perform fairly well in the robust case without explicitly taking into account robustness; their average sub-optimality with respect to Robust is no bigger than 6%, for example. On the other hand, in addition to its increased computational requirements, Robust suffers from larger sub-optimality with respect to both Basic and MaxGap.

4.2 Specific instances

It is also important to note that the trends just described are not uniform across all instances. For example, using a red bubble in each plot of Figure 4, we highlight an instance for which Basic and Robust return the same solution. Table 1 reports the objective values of the various methods for this instance, and Figure 5 gives another perspective on the same.

	Objective		
Method	Basic	MaxGap	Robust
Basic	1	0.3259	18
MaxGap	2	0.8680	19
Robust	1	0.3259	18

Table 1: Objective values for the three methods for an instance represented by the red dot in Figure 4. See also Figure 5.

⟨tab:good_us⟩

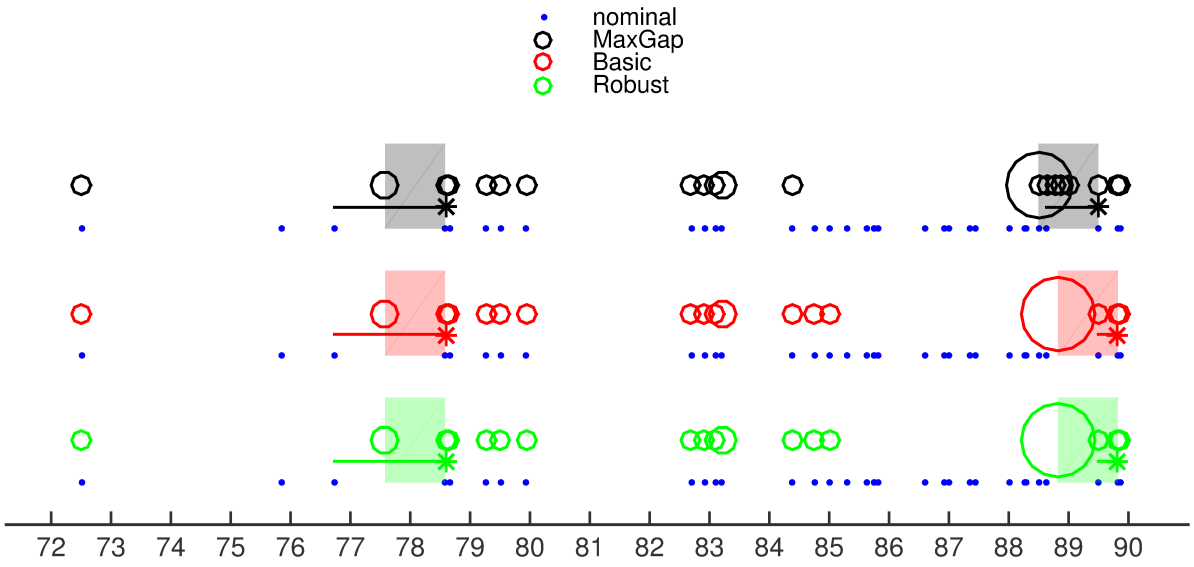

Figure 5: Worst-case grades for the solutions (a, b) of the three methods for an instance represented by the red dots in Figure 4.

⟨fig:good_us⟩

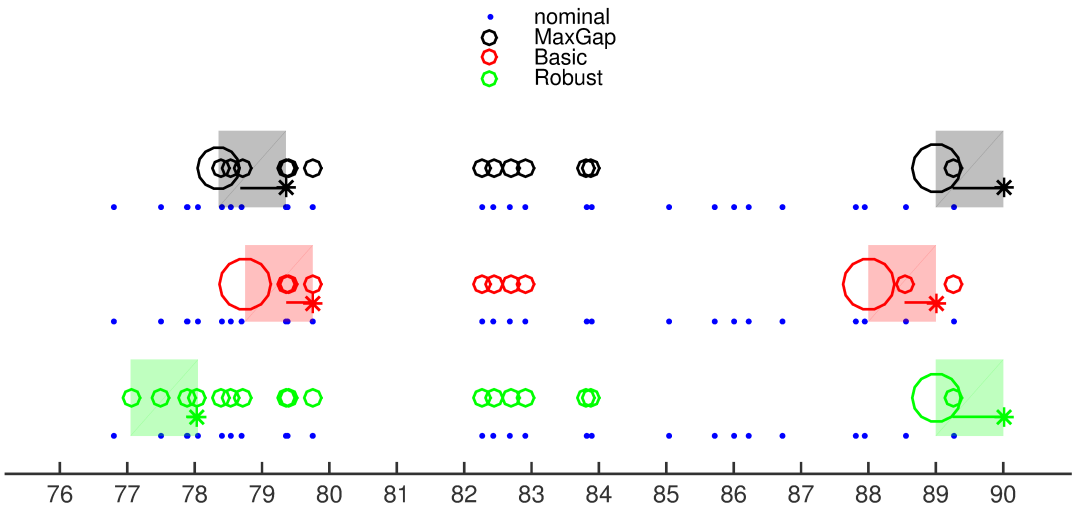

Figure 6: Worst-case grades for the solutions (a, b) of the three methods for an instance represented by the yellow dots in Figure 4.

⟨fig:good_mg⟩

Figure 5 can be viewed as several bubble plots—one for each method, Basic, MaxGap, and Robust—plotted on the number line. Each plot depicts, for the corresponding method's solution (a, b), the worst-case realization of $f = Gw$ over all G. That is, given the method's solution (a, b), the plot shows the final grades f which, if realized, would cause the maximum number of pseudo-borderline students. The larger the bubble, the more students having the plotted grade f_i. Additional information is shown in each plot for reference. The nominal grades are shown as blue dots, and the shaded rectangles in each plot represent the ranges for borderline students for a and b, respectively. Each rectangle is width $\epsilon = 1$, and note that the right-most edge of the rectangles are precisely a and b, which are marked by asterisks. Finally, in each plot, the two solid lines illustrate gap(a) and gap(b) by showing the distance between a and b and the first nominal grade to the left, respectively.

Finally, in Table 2 and Figure 6, we examine an instance—indicated by the yellow dots in Figure 4—for which MaxGap outperforms both Basic (relative to Robust) and Robust (relative to Basic).

	Objective		
Method	Basic	MaxGap	Robust
Basic	3	0.3715	21
MaxGap	4	0.6579	18
Robust	5	0.1538	14

Table 2: Objective values for the three methods for an instance represented by the yellow dots in Figure 4. See also Figure 6.

⟨tab:good_mg⟩

5 Conclusions

In the paper, we have introduced three strategies for computing the breakpoints (a, b) for assigning final letter grades. The first one (MaxGap) maximizes the distance between the breakpoint and the first nominal grade below it. The second (Basic) finds the optimal breakpoint with respect to the number of pseudo-borderline students in the nominal case. Finally, the third (Robust) minimizes the maximum number of pseudo-borderline students over the uncertainty set. Results on realistic instances show that the MaxGap and Basic strategies are quite robust themselves, even though there are some cases for which explicitly computing the robust solution is worthwhile. In general, the practical choice would be for a professor to consider the three alternative strategies in a particular course. This is viable since the computational times are not too high for a moderately sized course. It could also be useful to look at the worst case f, as depicted in Figures 5–6, to judge whether that particular f seems likely or not. If not, then the uncertainty set could be adjusted.

If computation times become a bottleneck, particularly for Robust, it may be possible to decrease the size of the mixed-integer representations (3) and (6) using an approach described in [12] for compact formulations of the union of polyhedra.

Veronica, should we acknowledge the financial support that paid for my Rome visit in 2014?

References

emirovski.2009 [1] A. Ben-Tal, L. E. Ghaoui, and A. Nemirovski. *Robust Optimization*. Princeton University Prss, Princeton, N.J.,, 2009.

mas.et.al.2011 [2] D. Bertsimas, D. B. Brown, and C. Caramanis. Theory and applications of robust optimization. *SIAM Review*, 53(3):464–501, 2011.

.Sendhoff.2007 [3] H.-G. Beyer and B. Sendhoff. Robust optimization: A comprehensive survey. *Computer Methods in Applied Mechanics and Engineering*, 196(33-34):3190–3218, 2007.

bko.et.al.2005 [4] P. Bobko, P. L. Roth, and A. Nicewander. Banding selection scores in human resource management decisions: Current inaccuracies and the effect of conditional standard errors. *Organizational Research Methods*, 8(3):259–273, 2005.

vey.et.al.1995 [5] J. M. Mulvey, R. J. Vanderbei, and S. A. Zenios. Robust optimization of large-scale systems. *Operations Research*, 43(2):264–281, 1995.

er.Wolsey.1988 [6] G. Nemhauser and L. Wolsey. *Integer and Combinatorial Optimization*. John Wiley & Sons, 1988.

zer.Healy.2012 [7] S. Rojstaczer and C. Healy. Where a is ordinary: The evolution of american college and university grading, 19402009. *Teachers College Record*, 114(7):1–23, 2012.

ett.et.al.2001 [8] P. R. Sackett, N. Schmitt, J. E. Ellingson, and M. B. Kabin. High-stakes testing in employment, credentialing, and higher education: Prospects in a post-affirmative-action world. *American Psychologist*, 56(4):302–18, 2001.

Schmidt.1995 [9] F. L. Schmidt. Why all banding procedures in personnel selection are logically flawed. *Human Performance*, 8(3):165–177, 1995.

emhauser:2010 [10] J. P. Vielma, S. Ahmed, and G. Nemhauser. Mixed-integer models for nonseparable piecewise-linear optimization: Unifying framework and extensions. *Operations Research*, 58(2):303–315, 2010.

Vielma2008467 [11] J. P. Vielma, A. B. Keha, and G. L. Nemhauser. Nonconvex, lower semicontinuous piecewise linear optimization. *Discrete Optimization*, 5(2):467 – 488, 2008.

Vielma2011 [12] J. P. Vielma and G. L. Nemhauser. Modeling disjunctive constraints with a logarithmic number of binary variables and constraints. *Mathematical Programming*, 128(1):49–72, 2011.