Teaching Portfolio

Contents

1 Professional Information

1.1 Statement of teaching goals and professional reflection

As mentioned in my statement of teaching philosophy, I strive to ensure (1) access to knowledge and to students and (2) excellence in my teaching skills and students' acquired skills. Now, these goals being stated, my approach to teaching has evolved over the years, slowly morphing into what it is right now. I want to hope that it is still in flux because teaching should always evolve. Over the last few years, my teaching and the activities I provide outside the classroom have been guided by the realization that I may have forgotten to acknowledge our students' abilities.

My point of view has changed: I do not come to class anymore only to "teach" students content I bring. Instead, I come to class planning to make them realize how much they know already. A large part of my work then consists in building their confidence in skills they have but are not aware of. Once this is done, I can then work with them to take them a few steps further. What that means is that I do not position myself as their "fixer", someone who will fix the problems of our "unprepared" students: instead, I acknowledge their background and however "unprepared" one might think they are, I believe and share with them repeatedly that they already know a lot, that they can do the work, and I demonstrate that to them. I find that, all the more in our institution where students come with a wide range of demonstrated skills, it would be contradictory to our mission of access to accept them in the classroom only to remind them and make them feel unfit, unprepared, while, when looking closely, they are not at all unfit, they just come with a possibly non-traditional set of skills, but still with skills. This attitude is now a central part of my philosophy.

For instance, in CS1, the intro to computer science course (a.k.a., CS1401 or CS1301/1101), the goal is for the students to become proficient at solving problems, and along the way, to learn how to implement their solution to problems in some programming language (so far, it is mostly focused on Java, but it could be anything). I insist with my students on the fact that they already know most of what I am going to teach them: they solve problems everyday, they use conditionals, they use loops, and even recursion, methods. We spend quite a while "excavating" these daily habits via hands-on activities so that students realize that, indeed, they do all that already. Then strong on their new confidence in the subject matter, I help them structure their approach, challenge the way they present their solutions (namely, algorithms), etc. We are on our way to progressing a few steps further together. This is how I see acknowledging my students' prior skills and abilities.

In addition to building on my students' existing abilities, my teaching style has also evolved in regards to assessing my students' work and final grade. First of all, let me start by stating that I believe that students should be able to compute their current grade throughout the semester to have an indication of their performance: it is really important for them, all the more for entering students, whom I have taught primarily over the last 3 years. As a result, I share with them the relative weight of every type of grade they will get during the semester (this is what is in the syllabus). However, I share with them that the grade they can compute will only be a lower bound of their actual final grade. The reason is that I also share with them that I am not interested in grading their speed of learning but rather what they become, whether they will be ready for the next course in sequence or not. In doing this, I set the atmosphere in the class to be one where failing is ok, where taking time to learn is ok, because not understanding something only means that a student does not understand it yet, not that he or she will not understand it. As a result, my way of assessing students is very much based on competencies rather than semester-long performance. I find this way of doing to be much truer to my belief that students come with different backgrounds and learning styles and should not be penalized for starting from farther or taking longer to learn a concept, as long as it is acquired by the end of the semester. A turning point in my attitude towards grading and students' learning was when I attended a presentation of Dr. Carol Dweck, from Stanford University, at the NCWIT Summit in 2014.

In what follows, I provide details about the courses I taught since 2012, material I have developed, technology I use or have used in the classroom, and the students I have advised.

1.2 List of courses taught, Teaching load information, including level and class size

1.2.1 Overview

Since fall 2012, I have taught the following courses:

- at the **undergraduate** level:
  - CS1190: Topics in Computer Science – Problem-Solving and Algorithms (once)
  - CS1401, later CS1301/1101: Introduction to Computer Science (6 times)
  - CS2401: Elementary data Structures (once)
  - CS3350: Automata (4 times)
  - CS3360: Concepts and Implementation of Programming Languages (3 times)
- at the **graduate** level:
  - CS5303: Logical Foundations of Computer Science (once)
  - CS5350: Advanced Algorithms (once)
- **cross-listed** courses (UG and G):
  - CS4365 / CS5354: Topics in Intelligent/Soft Computing – Problem-Solving and Algorithms (once)

Courses numbered CS1XXX, CS2XXX, CS3XXX, or CS4XXX are undergraduate-level courses, CS3XXX and CS4XXX being upper division courses. Courses numbered CS5XXX are graduate-level courses.

1.2.2 Independent studies

Since Fall 2012, I have offered 3 independent studies at the undergraduate level (CS4371) and 3 at the graduate level (CS5391), including one at the doctoral level (CS6391).

During this time, I have also supervised a number of students, as illustrated by the number of graduate research and dissertation "courses" I was in charge of:

- Computer Science Research: CS 5399 (Master's thesis, spring 2013 – 3 students, fall 2014 – 2 students, spring 2015, summer 2015), CS 6394 (doctoral research, spring 2017, fall 2017), CS 6694 (doctoral research, spring 2017, fall 2017), CS 6399 (dissertation, fall 2012, spring 2013),
- Computational Sciences Research: CPS 5396 (Graduate Interdisciplinary Research, fall 2013), CPS 5397 (graduate research, summer 2014, fall 2017), CPS 5398 (Graduate Thesis, fall 2014), CPS 5399 (Graduate thesis, spring 2015, summer 2015, fall 2015), CPS 6396 (graduate research, fall 2012, spring 2016, fall 2016), CPS 6397 (doctoral project, fall 2012, summer 2016, fall 2016), CPS 6398 (dissertation, fall 2016), CPS 6399 (dissertation, fall 2012, spring 2017)

1.2.3 Lectures

My teaching load since Fall 2012 in terms of lectures is reported in Table 1.

Term	Subject	Course Title	Enroll.
Fall 2012	CS 3360	Concepts and Implementation of Programming Languages	29
	CS 5350	Advanced Algorithms	17
Spring 2013	CS 3360	Concepts and Implementation of Programming Languages	29
	CS 5303	Logical Foundations of Computer Science	11
Fall 2013	CS 3350	Automata	40
	CS 3360	Concepts and Implementation of Programming Languages	42
Spring 2014	CS 2401	Elementary Data Structures	24
	CS 3350	Automata	36
Fall 2014	CS 3350	Automata	47
Spring 2015	CS 1401	Introduction to Computer Science	42
	CS 3350	Automata	48
Fall 2015	CS 1401	Introduction to Computer Science	39
Spring 2016	CS 1401	Introduction to Computer Science	47
Summer 2016	CS 4365	Topics in Soft Computing	10
	CS5354	Topics in Intelligent Computing	5
Fall 2016	CS 1401	Introduction to Computer Science	45
Spring 2017	CS 1301	Introduction to Computer Science	45
Fall 2017	CS 1301	Introduction to Computer Science	51
	CS 1190	Topics in Computing: Problem Solving	10
	CS 1190	Topics in Computing: Problem Solving	N/A

Table 1: List of lectures from fall 2012 to fall 2017

Teaching Load Information: Usually faculty in my department teach 2 courses a semester (or 6 credit hours). My teaching load was lower in fall 2014, fall 2015, spring 2016, fall 2016, spring 2017, and fall 2017 as I was released from one of my two courses. Also, despite my course release, I have always had a reasonably high number of students, average about 47 students per semester on semesters with a course release, and about 64 on regular semesters (excluding summer when enrollment is typically lower).

1.3 New courses and/or major course revisions

Since fall 2012, and specifically since fall 2014, I redesigned our department's introduction to computer science course several times (2 iterations with major revisions), I created two new courses: one cross-listed 3-credit-hour topic course on problem-solving and algorithms, and a new one-credit-hour course on problem-solving and algorithms for undergraduate students only.

1.3.1 Redesigns of Introduction to Computer Science (CS1401, later CS1301 and CS1101)

Redesign No. 1. This redesign brought to the course a few new elements: a somehow flipped classroom instruction, an online textbook with automatic grading of embedded exercises, a synchronized instruction (across the 3 sections of CS1), and a few new topics introduced in the class.

In fall 2014, David Novick, Monika Akbar, and myself (who were scheduled to each teach one section of CS1401 in spring 2015) were tasked to revisit the way CS1401 was taught. Together we moved this course to a **flipped course**. Flipped courses are courses where most of the content is delivered outside of the classroom, freing time in class for hands-on activities, practice, and deeper discussions. In fact, using textbooks, which is a very traditional way of teaching, was already a way to prepare students before they come to class. However, the realization that students were not necessarily doing that gave way to the flipped classroom "novel" idea. In fact, most of what there is with that idea is that videos, rather than or in addition to textbooks, are provided for students to watch outside of the classroom, and which explain topics to be covered in class. We designe videos, starring David Novick with the goal that students would watch them before class and come prepared so that we could focus on activities and discussion in class. In fact, I observed that students were still not coming to class very prepared (for many reasons including the lack of time due to many outside personal constraints). As a result, I ended up not teaching a fully flipped class but one where, wile still giving more space to hands-on activities, I lecture a little bit: usually more at the start of the semester and less as we progress through the semester and students have already reviewed enough material (for midterm exams and/or lab) and can now focus on activities in class.

We changed the textbook and adopted an **online textbook** (zybook: Programming in Java – see appendix). After review of this book, it had appeared to us that it covered most of what we wanted the students to read for the class. This textbook (which we are still using) has a few very practical features, including: 1/ you can modify it (move the sections around, remove some, set some as optional, etc.), 2/ it has embedded activities – participation and challenge activities, 3/ we can keep track of the progress of each of our students of these activities. This last feature allowed us to be more aware of our students stuggles or study problems: as soon as we see students not completing their homework, we can contact them or talk to them in class to figure out how to put them back on track. That is an essential element of our pedagogy: making sure that we know our students, that they know we look at their performance individually, and that we are there to help them.

What we also did in this first redesign was to **synchronize our teaching** (not our teaching style) so as to ensure that all students, across all 3 sections, would get the same set of outcomes met and relatively at the same time because we share students across 6 lab sections. As a result, we were also able to provide the lecture plan as well as deadlines of labs and homework (see appendix) to students much ahead of said deadlines so that they could better organize their time (a lot of our students work out of campus, some full time, so it is important that we provide time for them to plan their work).

Finally, we **revisited the outcomes** of the course (see appendix). We kept all the outcomes of the previous version of this course and we added a few more: namely, multi-D arrays, recursion, and linked-lists. The rationale behind these additions was that we wanted to expose the students to some key topics of CS2 (a.k.a., CS2401: Elementary Data Structures) at a level 1, because I had observed, while teaching CS2, how students would be frightened by the novelty of these topics when these topics were in fact not hard, but just had to be introduced in context in CS1 to provide students with the big picture of storage for instance or repetition. So instead of giving them the impression that "we are only going to do repetitions through loops and then, you'll see something more complicated called recursion", we introduced both concepts at the same time. Instead of telling them that "they could only store data in 1D arrays", we introduced the concept of multiD arrays right away, while providing more practice on 1D and some on 2D. The result of this was that students showed less fear when covering these topics in CS2 and performed better.

Redesign No. 2. In spring 2016, I received funding from Google to further the changes effected in CS1. This second redesign was not major but **focused on providing more tailored service to students**. Namely, I started using an online quzzing system, Socrative.com, to be able to assess my students more often. I started Saturday sessions with my students who wanted to catch up or go further than what we were studying in class. I used Google EngageCS resources (labs reviewed by educators) and I also worked on labs that I could contribute to Google EngageCS.

Redesign No. 3. This last redesign came at the time when we also splitted CS1401 into CS1301 and CS1101. This split was motivated by the fact that students often fail the course because of poor performance in labs and we wanted to allow them to retake only the lab part (CS1101). This is to acknowledge that our students face unique constraints and challenges outside the classroom, having to work full time or supporting a family, their parents. Because of this, we often observed that students were struggling to meet the deadlines for labs, or even to complete the labs, making them fail both the lecture and the lab part of CS1401. By splitting it into a lecture and a lab, we offered more flexibility to our students. Aside from this note, the changes brought by this last redesign, sponsored by UTEP's STEM-Accelerator program, consisted of: a **new sequence of topics** and a grading style focused on **competencies**.

Overall, the topics I covered in CS1 remained the same[1] What I did, however, was that I designed a **"layered" plan of instruction** where I first teach all topics (except objects) at a high level, trying to make them connect their daily experiences to all the topics covered in CS1 (there are very

[1] We did have to somehow modify the outcomes of the course because of the split into CS1301 and CS1101. However, overall, the outcomes remained the same when looking at the two courses together.

natural examples of that), and then I came back to these topics more in depth. My semester is therefore now organized in 4 phases (see appendix): Phase 1 – general coverage of all topics. This phase is crucial to their realization that they already know most of what we are going to cover during the semester. For instance, I put a lot of emphasis in connecting all I present to situations in their daily lives. Phase 2 – tinkering: we start using memory, conditionals, repetitions, methods, but mostly from given code and algorithms. Phase 3: doing – the students can now implement their own solutions to problems, we also dig deeper into repetitions with recursion. Phase 4: creating – while still using all we've learned so far, we go over objects and conclude with some implementation and the use of linked-lists. This "layered" approach allows me to go over concepts several times during the semester. It allows students to have more time to acquire skills.

This brings me to my other change: I now focus on **competencies**. As our students come with varied levels of what is usually understood as academic preparedness, and since I feel very strongly about our mission of access, I believe that penalizing students for the level they come at (which might be reflected in lower grades at the start of the semester) is counterproductive and only contributes to "confirming" to them that computer science or even higher education is not for them. On the contrary, I believe that our students come with a lot of potential and experiences that we can leverage to make them successful (this is a big part of my Phase 1 – see above). I want them to realize that 1/ they bring a lot to their success and 2/ they can grow even further. I try my best to apply the advice of Dr. Carol Dweck (from Stanford University) about the growth mindset. As a result, in the classroom, I share with students my philosophy about their potential, my belief that they can and will grow, and my intentions to work with them to make this happen. To be truthful to that, I also emphasize that I value failure, that it is part of the learning process, and that I am mostly interested, when it comes to assessment, about what they will be like when they leave my class to move on to the next one. Quizzes and exams are no longer potential penalizing instruments, but assessment instruments that inform us (the students and myself) about what we should do next, individually, to meet the goals of this course.

1.3.2 Problem-Solving for CS Students

A few semesters ago, as I was teaching CS1, I realized that our students struggled particularly with solving problems. This was surprising to me because I trust that they are able to solve the simple problems I had given them. I wanted them to realize that they were skilled problem solvers and up their confidence in this respect because I really believed that they were able to solve the problems at hand, just were experiencing anxiety and lack of confidence.

As a result, I started an informal problem-solving club for my CS1 students, later on extended to the CS2 students as well. Each week, for an hour, I met with students to "play" with problems, be challenged, and realize that we were able to address problems. It is important to note that most of the time, we did not use programming to solve these problems. The aim of this club was to boost my students confidence in understanding, attacking, and solving problems. I stopped holding this club in spring 2017 because of other commitments, but in the meantime, problem-solving has morphed into regular courses that I have created, as described below.

- **Cross-listed 3CH Course on Problem-Solving and Algorithms (CS4365/CS5354)**

  Based on the problem-solving club I was holding mostly for CS1 students, I wanted to extend this opportunity to more students. As a result, I proposed to offer a summer course on Problem-Solving and Algorithms (see syllabus as one the appendix documents). It was a cross-listed course offered in summer 2016 and attended by 15 students (10 undegraduate students and 5 graduate students). Offering this course allowed me to think about problem solving in a more structure way, to identify material and outcomes relevant and valuable to the students. This course received good evaluations and was the basis for my next contribution (see below).

- **New 1CH Course on Problem-Solving and Algorithms (CS1190)**

  As our CS department has embarked in our NSF RED project, our faculty have brainstormed about how to provide an even better experience for our students. Among many potential aspects of this problem, one that I am familiar with as a CS1 instructor, is attrition in early semesters and lack of confidence in problem-solving skills. Attrition in early semesters is often linked with our students' lack of sense of purpose, of understanding of CS' "big picture". This is reinforced by the fact that in their first three semesters in CS, our students typically[2] only have one computer science course available to take each semester (namely: CS1, CS2, amd then CS3). As a department, we proposed to offer 1-credit-hour courses available for our CS students to take in parallel of their "thin" sequence CS1-CS2-CS3. One of the courses we proposed to offer is a 1CH course on Problem-Solving.

  In spring 2017, our department, led by Ann Gates, started working with Google Engineers, Lorne Needle and Mike Gainer, on the creation of problem-solving courses, in collaboration with CAHSI institutions, New Mexico State University and Universityof California Dominguez Hills. In this collaboration, we came up with the idea of three 1CH courses on problem-solving, from intro to advanced level. I was in charge of creating the intro course, which I taugh as a topics in computing CS1190 over 6 weeks (2.5 hours per week) in early fall 2017 and am set to teach again in late fall 2017. In late summer 2017, while finishes the preparation of the pilot of this CS1190 course, Johannes Strobel (from University of Missouri) joined the team

  My problem-solving course focuses on developing the students' ability to attack a problem, any, even outside of the students' area of expertise. I provide a problem-solving framework so that they have a structured approach and are aware of important steps in problem solving. A lot of what I do in this course is helping the students realize how skilled they are and helping them reflect on their problem-solving approaches so that we can understand them and refine them. When I taught it as a 6-week course in early fall 2017, with 10 students, students were exposed to a variety of problems, from short riddles to larger problems, some proposed by Google engineer Mike Gainer, and one by Craig Tweedie, an environmental scientist, professor at UTEP. The culmination of this course was a project presentation to Google engineer Lorne Needle. So far, feedback has been very positive. The course was observed and videorecorded throughout the 6 weeks. Interviews are being conducted by the education researchers who were observing the course and we expect to receive more constructive feedback within a few weeks. I have debriefed with Lorne Needle, Ann Gates, Christina Convertino, and Johannes Strobel and am working on a refined version of the course to be taught in the second half of the fall 2017 semester.

[2] Under some conditions, students taking CS3 can also take a few upper-division CS courses at the same time. However, that still leaves them with two semesters with one CS course per semester, and not all of our students are eligible to take additional CS courses while taking CS3.

1.4 Evidence of curriculum development, including sample syllabi and course materials

1.4.1 Changes in CS1401: Introduction to Computer Science

All information provided below is supported by evidence available as appendix to this document[3].

□ **Modification of the lecture plan and synchronized teaching.** Each semester I provide a plan for my students. I am providing, as evidence of changes, such a plan from spring 2015 and from fall 2017.

□ **New outcomes.** In spring 2015, we modified the outcomes of CS1401. In fall 2017, we modified them again to account for the split into CS1301 and CS1101.

□ **Online textbook.** We use Zybook Programming in Java. I modify the book each semester to best fit my teaching style, content, and sequence.

□ **Online quizzing system.** I have designed many quizzes for CS1401/CS1301. They are available on piazza, as I make them available to the students for practice. They are also shown as a list as appendix.

□ **Course material.** Every semester, I work on improving my lecture notes (and slides) and I work at reinventing lab topics. Therefore, over the six times I taught CS1401/CS1301, I have, in collaboration with our instruction team (other instructors, TAs), significantly contributed to the material that can be used for teaching this course.

1.4.2 CS4365/CS5354: Topics in (Soft/Intelligent) Computing – Problem Solving and Algorithms

I created this course from my experience holding a problem-solving club. I created a syllabus, outcomes, lecture notes, activities material, exams, and projects for this course. I provide as evidence the syllabus and exam for this course.

1.4.3 CS1190: Topics in Computing – Problem Solving

I created this course based on my prior experience with the problem-solving club and CS4365/5354 on problem-solving. Creating this course required writing a new syllabus, preparing a repository of possible activities, a day-to-day plan of activities, homework, and exams. I provide as evidence the syllabus, repository of activities, and exams.

1.4.4 Other efforts

In addition to the above efforts, I have developed some material integrating my research in teaching, which I have used in courses such as Artificial Intelligence when I was guest lecturing. I developed lecture slides and labs for my CS1 course. I developed labs and projects for my CS3360 and CS3350 courses, and projects for my CS5350 course.

[3] Supporting documents are listed in the appendix of this document and available on the webpage I created for this dossier.

1.4.5 Evidence of teaching style evolution

As mentioned in my teaching philosophy and in my statement of teaching goals and professional reflection, making my teaching style evolve is part of my philosophy. Now, more specifically, over the past few years, my main teaching style changes can be summarized in the following list:

- **I share with students my growth mindset.** A consequence of that is that I now focus on **acquisition of competencies/skills**, rather than possibly penalizing students for slow beginnings or slower learning pace.
- **Use of online activity resources:** over the years, and in the aim to both better serve our students and to address a growing enrollment, I have turned to systems that automatically grade my students' work (hence, they get feedback faster) and hence allow me to assess them more often (so I know their struggles better and can better help them). I now use Zybook (an online textbook) when relevant and Socrative.com to conduct quizzes regularly.
- **Enforcing student engagement in CS1 as part of the grade:** 5 points in the CS1 overall grade acknowledge students' engagement in the computing community. They can participate in outreach activities, attend seminars, join a research group, etc. (see list on my CS1 syllabi, in appendix). I believe, and that was a joint decision of the CS1 instructional team, that enforcing such practices further contributes to engaging students in their major and to building their identity as CS students.

1.5 Evidence of use of technology to complement instruction

I always look for ways to improve our students' experience. Over the years, as mentioned earlier, I have polished my style as an instructor and I tried technology tools with varied success rates. In what follows, I describe the tools I use to support my teaching.

1.5.1 Online Forum and Wiki: Piazza

Piazza can be seen as an alternative to BlackBoard. I have used it since 2010 and I have stayed with it as I find it intuitive and light. I create one piazza "site" for each course I teach and I enroll on it all students enrolled in my course as well as the TAs and other assistants and possible peer leaders. It is a combination of a forum (students can interact with each other, ask questions, answer them, ask questions to the instructors, instructors can answer any students' questions, etc. – it is very flexible) and a wiki (I post all information and material pertaining to the course on its piazza "site".

Piazza can be integrated with BlackBoard to allow linkage with students' IDs and grade books. However, the link is not yet implemented at UTEP. That is currently a drawback of my use of piazza, but one that does not counterweigh its advantages. The links to my

1.5.2 Online Quizzing System: Socrative

Since fall 2015, I have regularly used Socrative.com. Socrative is an online quizzing system. It allows me to create quizzes for my students and have these quizzes automatically graded. It also allows me to conduct short questions during lectures as I see fit. I also use the "short question" fonctionality to leave a stream of questions open during the lecture so that students who may be shy to ask questions in person can ask them directly on Socrative. When I see that questions are being asked, I then address them to the whole class.

I now use the pro version of Socrative in which I can use several quizzing "rooms" (so that I can have several quizzes open at the same time: for instance, one open for a take-home quiz, and one in lab at the same time). It also allows me to identify my students by ID (which I design and give them at the start of the semester), which helps me process grades faster and identify faster who was not present for an in-class quiz for instance. I also use Socrative once at the start of the semester and once at mid-term to survey my students: at the start of the semester, I use it to learn more about my students. At mid-term, I use it to check their satisfaction and/or struggles so far in the semester, so that I get a better understanding of my class dynamic.

1.5.3 Online Textbook: Zybook

Since spring 2015, I have used an online textbook for CS1. Our instructional team at the time (myself with David Novick and Monika Akbar) decided to use the Zybook Programming in Java online textbook. It covers all the content we need the students to read about. Of course, it is focused on Java, while our CS1 course aims to be a course in which we teach problem-solving with computers rather than a specific language. However, in practice, this course has always been taught with a focus on one language (and exposure to a few more). Currently our department has Java as its first language, so we thought that Zybook Programming in Java, although with a strong focus on programming, was a suitable textbook.

We find this online book convenient as it lets us modify the book as we see fit for our class: in particular, we can change the order of chapters, we can create new chapters out of sections of existing chapters, we can remove content, make some content optional, etc. In addition, participation and challenge activities are embedded in the book, allowing the students to practice. Now, such features are usually common among online textbooks. What decided us on this online textbook rather than another one, and frankly, on an online textbook altogether, was the fact that we can follow our students' progress on their assigned embedded activities. As a result, we have immediate feedback on the students' diligence to complete homework and/or struggles to do so. Whatever the reason may be, we are immediately aware that something is not right and we can contact students to figure out how to help them catch up.

So far, I have used this online textbook only for CS1 since I have not taught another course for which they have an online textbook. Also, I have met with representative of other publishers and tried other online textbooks. Some look very good and their number of exercises is very appealing. However, I have not yet found one that has a good balance of exercises and allows to track the students' progress.

Note: I used an extra feature of Zybook during Fall 2016 called Zylab. It is a lab, embedded as extra sections in the chapters of the existing Zybook. We, as instructors, design our own labs and they are added to the book. They come with automatic grading. They allow students to test their work as many times as they want before submission and we can decide whether to allow one or multiple submissions. In this regard, this tool was very practical. However, I ran into some problems when using it: 1/ I was not able to assign I/O labs to my students as there was no way to read or write files. The work to be done by students had to consist of one file and terminal execution only. 2/ It did not support multi-D arrays well. 3/ for the same reason as what prevented me from assigning I/O labs, I could not assign work with objects sine only one file could be handled per lab. As a result, for these labs, I had to revert to traditional labs, not within Zylab and I opted out of this option the following semester as I did not want students to have to pay extra for an option that was not fully functional.

1.5.4 Online Programming Resources: CodingBat.com

In addition to the online textbook I use for CS1, I also make use of other programming resources as pointers for the students to practice more. In particular, one that I like for the variety of its problems and rigor in their description is codingbat.com. I have used it for several semesters in CS1.

1.6 Professional development in teaching, including workshops and seminars presented and attended

Always seeking innovation in the classroom to increase student success, break the barrier of usual intimidation, engage all students, I have tried several approaches over the years, from using the old hotmail messenger back in the days to answer my students' questions in real time, to being an early adopter of piazza (piazza.com) to engage students through active participation and to still allow one to one mentoring, to online textbooks and labs (zybooks.com including zylabs) and online quizzes and online quick feedback system Socrative (socrative.com).

I have also regularly sought professional development (see below) and integrated what I learned in my classes: cooperative learning and problem-based teaching, flipped classroom, competency-based assessment and motivation-based teaching.

- **Workshop on Teaching, by Olin College instructors**, organized by UTEP STEM-Accelerator Project team, June 2016.
  - Participation at this workshop led to my application to funds to help me redesign CS1. I did that in spring 2017.
- Presented a **webinar on Computational Thinking in the Classroom**, in April 2014, for all teachers of the Ysleta Independent District, El Paso.
- Presented **4 talks at the "Networking Technology & Content Conference"**, El Paso, TX. (Nov. 2014 and 2015), a regional teachers' conference: about integrating computational thinking in K-12 classrooms, and not only in math and science.
- **Continuing Education Program**, "Flipped Learning Brown Bag discussion," Center for Research in Engineering and Technology Education (CREaTE), University of Texas at El Paso. May 15, 2014.
- **Affinity Research Group training**: in summers 2011 and 2012.
- **Problem-Based Learning workshops**: in May 2012, May 2013, May 2014.

In addition, through the following events that I attended, I was able to glean some nuggets and tips about teaching:

- CRA-W Career Mentoring Working: Washington DC, November 2016.

- CE21 Community Meeting, NSF. January 2014.
- Workshop on Problem-Based Learning. 2013, 2014.
- Gender Summit: November 2013
- NCWIT Summer: May 2013
- Yearlong training: Leadership Development Institute at UTEP: 2012-2013
- NSF CE21 Community Meeting: 2011, 2012, 2014
- CRA Career: Washington DC, 2012
- CRA-W: Atlanta, 2012

2 Evidence of Teaching Quality

2.1 Student evaluations and comments, tabulated and summarized

2.1.1 Students' evaluations

In what follows, I report the students' evaluations of my courses (when available – not Fall 2017). All ratings are on a scale from 0 to 5.

Term	Subject	Course Title	Enroll.	Instructor	Course
Fall 2012	CS 3360	Concepts and Implementation of Programming Languages	29	4.44	4.08
	CS 5350	Advanced Algorithms	17	4.17	3.85
Spring 2013	CS 3360	Concepts and Implementation of Programming Languages	29	4.55	4.59
	CS 5303	Logical Foundations of Computer Science	11	4.1	3.9
Fall 2013	CS 3350	Automata	40	4.37	4.04
	CS 3360	Concepts and Implementation of Programming Languages	42		
Spring 2014	CS 2401	Elementary Data Structures	24	4.78	4.67
	CS 3350	Automata	36	4.63	4.34
Fall 2014	CS 3350	Automata	47	3.86	3.73
Spring 2015	CS 1401	Introduction to Computer Science	42	4.55	4.82
	CS 3350	Automata	48	4.0	4.0
Fall 2015	CS 1401	Introduction to Computer Science	39	3.4	3.6
Spring 2016	CS 1401	Introduction to Computer Science	47	4.12	4.06
Summer 2016	CS 4365	Topics in Soft Computing	10	4.5	4.33
	CS5354	Topics in Intelligent Computing	5	4.5	4.33
Fall 2016	CS 1401	Introduction to Computer Science	45	5.0	5.0
Spring 2017	CS 1301	Introduction to Computer Science	45	4.44	4.44

⋆ Students' evaluations and comments are available in appendix, Section 3. Note that the enrollment specified in the table does not correspond to the number of students who took the evaluation. Often, very few students did complete the evaluations.

2.2 Theses and dissertations supervised

2.2.1 Supervision of PhD students

1. **Paula Gonzalez Parra:** defended her Ph.D. in Computational Sciences in fall 2012. She worked on estimating best decisions to slow down and stop epidemics. She is now a professor of Mathematics at the University of Cali, Colombia.

2. **Xiaojing Wang:** fall 2009 to spring 2013. Xiaojing worked on Multi-Criteria Decision Making using fuzzy measures. She supported herself with her job on campus, which she still holds, as a research associate at our Center for Institutional Evaluation, Research, and Planning (CIERP). She graduated in spring 2013 and obtained a Ph.D. in Computer Science.

3. **Luis Gutierrez:** fall 2013 to fall 2015 – supported by **NSF CAREER grant.**
Luis was a long-term student in my research group, who started as an undergraduate when he was only taking the Introduction to Computer Science course. Due to medical issues, he had to drop out of the PhD program in the fall 2015. His contributions while a PhD student in my group are as follows:
Presentations:
□ El Paso, Texas, UTEP Graduate Research Expo, November 7 2013: "A constraint-based model for generating t-wise test suites".
□ El Paso, Texas, 14th Joint UTEP/NMSU Workshop on mathematics Computer Science, and Computational Science, November 2 2013, Interval-Valued Probabilities to Interval-Valued Possibilities: Case Studies of Interval Computations under Constraints.
□ San Antonio, Texas, SACNAS, October 3 - 6 2013, Constraint based model to generate t-wise test suites.
□ El Paso, Texas, COURI Symposium, April 20 2013, Generating minimal t-wise constrained test suites.
□ El Paso, Texas, 16th UTEP/NMSU Workshop on mathematics Computer Science, and Computational Science, November 3, 2014: Validated Solutions of ordinary differential equations and prediction of behavior (poster).
□ SIAM Seminar on Bash Scripting, February 2014.
□ El Paso, Texas, Bio-Informatics Symposium: "Interval-Valued Probabilities to Interval-Valued Possibilities: Case Studies of Interval Computation under Constraints". Luis C. Gutierrez, Salem Benferhat, Martine Ceberio, Vladik Kreinovich, Rebekah Gruver, Marianna Pena, Matthew J. Rister, Abraham Saldana, John Vasquez, and Janelle Ybarra (poster). April 2014.
□ El Paso, Texas, Bio-Informatics Symposium: INCaS a new constraint solving toolbox. Luis C. Gutierrez, John Vasquez, Martine Ceberio (poster). April 2014.
Articles in which he was a major author:
□ Gutierrez, Luis, Salem Benferhat, Martine Ceberio, Vladik Kreinovich, Rebekah Gruver, Marianna Pena, Matthew J. Rister, Abraham Saldana, John Vasquez, and Janelle Ybarra. "Interval-Valued Probabilities to Interval-Valued Possibilities: Case Studies of Interval Computation under Constraints", 14th Joint UTEP/NMSU Workshop on Mathematics, Computer Science, and Computational Sciences (2013).
□ Gutierrez, Luis C., Carlos Nieto, Francisco Zapata, and Martine Ceberio. "Constraint-based model to Generate twise Test Suites", 2013 SACNAS National Conference (2013).
□ Gutierrez, Luis C., Francisco Zapata, and Martine Ceberio. "Generating Minimal t-wise Constrained Test Suites", COURI Symposium Abstracts Spring 2013 (2013).

4. **Leobardo Valera:** fall 2014 to present – **supported by ARL AHPCRC grant and NSF CAREER grant.**
Leobardo has worked with me on integrating and quantifying uncertainty in our work on reduced-order modeling. He is set to defend his PhD dissertation in fall 2017. Below are some of his contributions:
Presentations, including posters:
□ El Paso, Texas, 16th UTEP/NMSU Workshop on mathematics Computer Science, and Computational Science, Reduced-Order Modeling
□ November 3, 2014: AHPCRC BootCamp, "Using Wavelet Transformations for Reduced-Order Modeling", DoD, Santa Cruz, California.
□ Leobardo Valera and Martine Ceberio. How to use Constraint Solving Techniques (ICST) to Predict Future Behaviors of Dynamic Problems. Army High Performance Computing Research Center (AHPCRC), Annual Technical Review Meeting Stanford University, Santa Cruz, CA USA. March 2016.
□ Leobardo Valera and Martine Ceberio. Handling Uncertainty and Large Dimensions in Nonlinear Systems of Equations using Interval Constraint Solving Techniques (ICST). Army High Performance Computing Research Center (AHPCRC), Annual Technical Review Meeting Stanford University, Santa Cruz, CA USA. March 2016.
□ Leobardo Valera, Martine Ceberio and Horacio Florez, Handling Uncertainty and Large Dimensions in Nonlinear Systems of Equations using Interval Constraint Solving Techniques. Army High Performance Computing Research Center, Research Management Boarding Meeting, Stanford, CA, January 2016.
□ Handling Uncertainty and Large Dimensions in Nonlinear Systems of Equations using Interval Constraint Solving Techniques Army High-Performance Computing Research Center, Research Management Boarding Meeting, Stanford, CA, January 2016.
□ Introduction to Pairwise Testing. Definition and Examples, 17th Joint UTEP/NMSU Workshop on Mathematics, Computer Science, and Computational Sciences The University of Texas, November 2015.
□ Validated Solutions of ordinary Differential Equations and Prediction of Behavior. 15th Joint UTEP/NMSU Workshop on Mathematics, Computer Science and Computational Sciences The University of Texas. November 2014.
□ Using Wavelet Transformations for Reduced-Order Modeling. Army High Performance Computing Research Center (AHPCRC), Annual Technical Review Meeting Stanford University, Santa Cruz, CA USA. August 2014.
□ Leobardo Valera and Martine Ceberio. "Interval Constraint Solving Techniques and Model-Order Reduction to Enhance the Solution of Dynamic Systems". INFORMS Annual Meeting. Nashville 2016.
□ Leobardo Valera, Martine Ceberio. Interval Constraint Solving Techniques and Model-Order Reduction to Enhance the Solution of Dynamic Systems. 19th Joint NMSU/UTEP Workshop on Mathematics, Computer Science, and Computational Sciences. El Paso, TX, November 2016.
□ Leobardo Valera and Martine Ceberio. Large Numerical Simulations: Needs, Challenges, Current Solutions How can Interval Computations help? NAFIPS 2016.
□ Leobardo Valera, Martine Ceberio. Reduced Interval Newton Method, 18th Joint UTEP/NMSU Workshop on Mathematics, Computer Science and Computational Sciences, New Mexico State University, Las Cruces, New Mexico, March 2016.

□ Using Interval Constraint Solving Techniques in Dynamic Systems Behavior Prediction, Constraint Programming and Decision Making (COPROD 2015), The University of Texas El Paso, November 2015.
Using Interval Constraint Solving Techniques in Dynamic Systems Behavior Prediction, 18th Joint UTEP/NMSU Workshop on Mathematics, Computer Science, and Computational Sciences The University of Texas, November 2015.
□ Using Regularization to Improve the Rate of Convergence in a Model-Order Reduction (MOR) Problem. Presented at the 22nd International Symposium on Mathematical Programming. Pittsburg, Pennsylvania, ISMP'2015, July 2015.
□ Using Regularization to Improve the Rate of Convergence in a Model-Order Reduction (MOR) Problem. The 17th Joint NMSU/UTEP Workshop on Mathematics, Computer Science, and Computational Sciences. New Mexico State University, Las Cruces, New Mexico, April 11, 2015.
Model-Order Reduction Using Cubic Spline Curve-Fitting. UTEP Graduate Student Expo. The University of Texas at El Paso, November 2014.
□ Model-Order Reduction Using Cubic Spline Curve-Fitting. 15th Joint UTEP/NMSU Workshop on Mathematics, Computer Science, and Computational Sciences. The University of Texas at El Paso, November 2014.
<u>Honors and Awards:</u>
□ Graduate Award for Academic and Research Excellence in Computational Science. UTEP
□ Banner Bearer at UTEP's Winter 2015 Commencement (No academic, probably it should be deleted)
□ Outstanding Paper Award. Joint Annual Conference of the North American Fuzzy Information processing Society NAFIPS'2015 and 5th World Conference on Soft Computing. For the paper: Martine Ceberio, Leobardo Valera, Olga Kosheleva, and Rodrigo Romero Model reduction: why it is possible and how it can potentially help to control swarms of Unmanned Aerial Vehicles (UAVs), Joint Annual Conference of the North American Fuzzy Information processing Society NAFIPS'2015 and 5th World Conference on Soft Computing, Redmond, Washington, August 17-19, 2015.
□ Outstanding Paper Award. Joint Annual Conference of the North American Fuzzy Information processing Society NAFIPS'2015 and 5th World Conference on Soft Computing. For the paper: Francisco Zapata, Octavio Lerma, Leobardo Valera. How to speed up software migration and modernization, Joint Annual Conference of the North American Fuzzy Information processing Society NAFIPS'2015 and 5th World Conference on Soft Computing, Redmond, Washington, August 17-19, 2015.
17/ Scholarship to attend the ACW conference.
Leo is about to graduate, in fall 2017.

5. **Angel Garcia Contreras:** spring 2015 to present – supported by **NSF CAREER grant & STEM Accelerator program.**
Angel previously worked with me on his Master's thesis (see below). As a PhD student, he is focused on optimization techniques to handle dynamic systems with uncertainty. Below are some of his main contributions:
<u>Presentations:</u>
□ Using Interval Constraint Solving Techniques in Dynamic Systems Behavior Prediction, Constraint Programming and Decision Making (COPROD 2015), The University of Texas El Paso, November 2015.

□ Global Optimization via Speculation. Constraint Programming and Decision Making (COPROD 2015), The University of Texas El Paso, November 2015.
□ El Paso, Texas, 18th Joint UTEP/NMSU Workshop on mathematics Computer Science, and Computational Science, November 2015, Insights into using continuous constrained optimization methods to solve black box mixed integer problems.
□ Comparison of Strategies for Solving Global Optimization Problems Using Speculation and Interval Computations. NAFIPS 2016.
□ Global Optimization via Speculation. UTEP NMSU Workshop Spring 2016.
□ Using Interval Constraint Solving Techniques in Dynamic Systems Behavior Prediction, Constraint Programming and Decision Making (COPROD 2015), The University of Texas El Paso, November 2015.
□ El Paso, Texas, 18th Joint UTEP/NMSU Workshop on mathematics Computer Science, and Computational Science, November 2015, Insights into using continuous constrained optimization methods to solve black box mixed integer problems.
Additional notable experience:
□ Taught a programming course at Stanford's Army High-Performance Computing Research Center Summer Institute in summer 2015 and summer 2016.

6. **Omeiza Olumoye:** summer to fall 2015 – supported by **NSF CAREER grant.**
Omeiza is not one of my PhD students. However, I am part of his PhD committee and in 2015, he joined my research team to contribute to our work on optimization and intervals.
Presentation:
□ Poster on Interval Newton Methods, 17th Joint UTEP/NMSU Workshop on Mathematics, Computer Science, and Computational Sciences The University of Texas, November 2015.

7. **Imran Abdullah:** joined my research lab in summer 2017. He is a Computational Sciences Ph.D. student and is currently supported by this program on campus. He is starting to work on predictions for large dynamical systems.

2.2.2 Supervision of Master's students

1. **Joel Henderson:** fall 2011 to summer 2014 – **supported by AAAS grant.**
Joel completed his thesis on the topic of Multi-Criteria Multi-Agents Decision Making using Argumentation Frameworks. During his work with me on this topic, he presented his work at the following international conference: Numerical Computations: Theory and Algorithms International Conference, June 2013, in Italy. He now works at ARL White Sands.

2. **Angel Garcia Contreras:** fall 2011 to fall 2014 – **supported by NSF CAREER grant.**
Before to enroll in the Ph.D. program, Angel completed his Master's thesis with me. His work was in optimization using speculations. During his Master's work, he presented at the following article: Garcia Contreras, A. F., Wang, X., Ceberio M., Bixler R., Gutierrez L., "Interval Optimization to Predict Software Quality Assessment Decisions", at INFORMS OS'2012. He also presented several times at the UTEP/NMSU workshop. In particular he presented at the 14th Joint UTEP/NMSU Workshop on mathematics Computer Science, and Computational Science (November 2013): "Insights into using continuous constrained optimization methods

to solve black box mixed integer problems". Along with Joel Henderson (Master's student), he also presented at SACNAS, October 3-6, 2013. In summer 2013, he worked with Patty Hough at Sandia National Lab in Livermore on thei DAKOTA project.
internship at SANDIA Livermore in summer 2013.

3. **Hima Kondepati:** fall 2014 – **supported by NSF CAREER grant.**
   Hima joined my lab in fall 2014 as a new student to our program and looking for a topic. She was interested in optimization and I trained her in fall 2014 on this topic and on research methods. By the end of fall 2014, she had decided that she wanted to switch programs to follow more IT-oriented track, without research.

4. **Troy McGarity:** summer 2015 to spring 2016, **Rebekah Gruver:** fall 2015, and **Salah Atiyeh:** fall 2015 – **supported by NSF CAREER grant.**
   These three students in the Master's of Software Engineering at UTEP contributed to my work on optimization and constraint solving by building a website (NumConSol) that allows to use the solver developed by my research students, hence allowing for broader dissemination. They also packaged our solver as standalone executables, tailored for Windows and Mac. They all graduated and are now working, respectively at HP, GoDaddy, and Exxon Mobil.

2.3 Thesis committees

Below are listed the students for which I was in the Master's thesis or project committee. My involvement in the work of these students varied vastly, but in general, I also advised them: meeting with them to review their work, propose changes and research directions.

1. **Chanel Perez**, fall 2014, in Geology at UTEP, work directed by Aaron Velasco;
2. **Omeiza Olumoye**, since 2015 (expected graduation date: fall 2018), in Electrical and Computer Engineering at UTEP, directed by Thompson Sarkodie-Gyan;
3. **Anastasia Volkova**, since fall 2016, graduated in fall 2017, in Computer Science at the University of Pierre and Marie Curie, France, directed by Christoph Lauter, Thibault Hilaire, and Jean-Claude Bajard;
4. **Leticia Miranda Mojica**, since spring 2017, in Leadership Engineering at UTEP, directed by Meagan Kendall;
5. **Nancy Avila**, since fall 2017, in Electrical and Computer Engineering at UTEP, directed by Homer Nazeran;
6. **Esthela Gallardo**, since spring 2017, defended her proposal in summer 2017 (expected graduation date: Fall 2018), in Computer Science at UTEP, directed by Patricia Teller;
7. **Paden Portillo** and **Oscar Chavez**, graduated in fall 2012, in Software Engineering (Master's).

2.4 Supervision of undergraduate students

I have coordinated my research group, CR^{2}G, Constraint Research and Reading Group, since 2004. Students at all levels, from high-school to PhD students to post-doctoral researchers, have been involved in this group since then. Involving students in research is a very important part of what I do: it increases access and channels students towards excellence in a very active and participative way. In additiona to graduate students, I have a large number of undergraduate students whom, I believe, benefit greatly from being involved in research.

In my research group, I value and encourage peer mentoring and follow the Affinity Research Group model[4].

The undergraduate students in CR2G are generally introduced to the research we do during our weekly meetings and the time they spend in the lab in between, shadowing our more senior students (not necessarily seniors in studies, but seniors in my group). They are then quickly involved in an on-going project that they like and my other students and I give the new students nuggets of work to carry out, so that it is not overwhelming. Based on this work, they can start tasting what research is like: in addition to completing their work, they have to present it either formally or not, discuss it, and propose the next steps in our weekly meetings. Like in my classes, I put special emphasis on building my students' sense of mastery and confidence, so that we can then progress several steps further.

The number of undergraduate students in my lab varies from semester to semester. Some have been in my lab for several years while others stay one or two semesters before they move on to something else. This is common as I invite students to my research group very early on (from when they take CS1) and their interest might change as they discover more about computer science. However, I believe that their involvement in research from early on is a great asset for them to be successful in class and in research if they choose this path. The following list of students is the list of students who have been most active in my group since fall 2012.

1. **Luis Gutierrez:** spring 2010 to summer 2013. Luis joined my research team in spring 2010 as he was starting his B.S. degree studies at UTEP. As an undergraduate student, he worked on a variety of problems during his almost 4 years in my team as an undergrad. He attended a summer school on constraint programming and logic at the University of Perugia, Italy, in summer 2013. He presented at many local and regional symposium, as well as at a couple on international venues. He was supported by COURI starting in fall 2011 and until the end of his undergraduate studies.

2. **John Vasquez:** fall 2013 and spring 2014 – **supported by NSF CAREER grant.**

3. **Enrique Martinez:** spring 2014 to fall 2014. As a musician, Enrique worked on constraints and music. He left my group when he graduated.

4. **Jason Holt:** fall 2014 – **supported by NSF CAREER grant.**

5. **Luis Alcantar:** fall 2014 to fall 2015. Luis was supported by COURI in summer 2015. He left my group shortly in spring 2016.

[4] The Affinity Research Group is a "cooperative learning approach to involving students with diverse backgrounds", see https://www.createspace.com/3374426 for more details.

6. **Cristian Ayub:** since spring 2015. In summer 2017, he was supported by CAHSI (Computing Alliance for Hispanic-Serving Institutions). In addition he was supported every semester by either a scholarship of UTEP's Campus Office for Undergraduate Research Initiative (COURI) or by UTEP's Louis Stokes Alliance for Minority Participation (LSAMP – summer 2016). Until summer 2017, he worked on combinatorial testing using constraint programming. While working on this topic, he led a team of undergraduate research students in a NSF-funded COURI program called UTEP Lens of the Market: during this 6-month-long program, he and his peers worked on the marketability of their work on combinatorial testing and its applicability to real-world industry problems, such as transistor design. Since summer 2017, he has been studying quantum mechanics to identify ways to solve them using our team's tools for dynamic systems. Over his years in my team, he has presented several times (at least twice a year: at the COURI annual symposium and at the UTEP/NMSU workshop).

7. **Gabriel Max Felix:** summer 2015 to summer 2017. During his time in my team, Max was a very active student who worked on combinatorial testing. he was part of the team led by Cristian Ayub working on the Lens-of-the-Market project. He stopped attending regularly because of the constraints of his job on campus.

8. **Elsa Gonzalez:** fall 2015 to summer 2016. She was involved in the early work of Cristian Ayub on combinatorial testing.

9. **Raul Mena:** fall 2015 to summer 2016. He was involved in the early work of Cristian Ayub on combinatorial testing. Raul changed majors to Business and left our group in fall 2017.

10. **Miguel Zamudio:** since spring 2016. Miguel was part of the team led by Cristian Ayub working on the Lens-of-the-Market project. He is now moving to doing research on Machine Learning with Olac Fuentes, but is still participating in my lab.

11. **Phillip Hassoun:** since spring 2016. Phillip was selected to participate in Stanford's AHPCRC Summer Institute for undergraduates in 2016. He has worked on a variety of problems since joining my research team. Lately, he has been instrumental in porting our algorithms to a mobile device to demonstrate their ability to run on limited computational resources.

12. **Enrique Salcido:** since fall 2016. Enrique joined my research group as a member of the Lens-of-the-Market participant, in the team led by Cristian Ayub. He is a Math student and needed a team to participate in this program. After the end of the program, he asked to remain a member of my research group and has contributed to our work on optimization.

13. **Joseph Gutierrez:** since fall 2016. Joseph was originally a Mechanical Engineering student but he later (in spring 2017) changed his major to computer science. He worked in my team on robust optimization and was supported by COURI in summer 2017.

14. **Daniel Villa:** since spring 2017. Daniel was selected to participate in Stanford's AHPCRC Summer Institute for undergraduates in 2017. item **Jorge Quinones:** since fall 2017. Jorge is currently studying our work on dynamical systems.

15. **Jesus Padilla:** since spring 2017. In summer 2017, he was supported by CAHSI (Computing Alliance for Hispanic-Serving Institutions). Jesus has worked on dynamic systems. In summer 2017, he worked on time uncertainty and dynamic systems' behavior prediction. He is now implementing an interval constraint solver from scratch to be used on a mobile device.

16. **Sairy Cohen:** spring 2017. Sairy joined my research group in spring 2017. She was very active in her work on visualization of our numerical results and implementation of optimization techniques. She left in summer 2017 to participate in a summer internship and has been busy since then.

17. **Alexa Cortes:** spring 2017. Alexa is a mechanical engineering student. She joined my group in spring 2017 but is no longer a member as she found it hard to combine with the constraints of her academic schedule.

18. **Stephanie Figueroa:** spring 2017. Stephanie joined my group in spring 2017. She is no longer a member as it was hard for her to combine taking classes, having a job on campus, and attending to research as well.

19. **Samuel Tinevra:** spring 2017. Samuel joined my group in spring 2017. He left in fall 2017 as he got a job off campus and finds it hard to combine his academic and professional constraints.

20. **Ricardo Alvarez Lopez:** since fall 2017. Ricardo is currently studying our work on dynamical systems.

21. **Jorge Quinonez:** since fall 2017. Jorge is currently studying our work on dynamical systems.

22. **Noemi Hernandez:** since fall 2017. Noemi actually is a former NEXUS high-school student intern in my lab. She spent summer 2017 in my lab as a high-school student and decided to join my team in the fall when she started her studies at UTEP. Noemi is currently working very actively with Ricardo and Jorge on our team's efforts on solving dynamic systems.

2.4.1 Other groups supervised

In addition to the above students, I have more informally advised the following three graduate students.

1. **Francisco Zapata:** was a Ph.D. student of Vladik Kreinovich who joined my lab during a year prior to his graduation. He contributed to our work on pair-wise testing.

2. **Quentin Brefort** (spring and summer 2015) from ENSTA, France, directed by Luc Jaulin; and

3. **Anthony Welte** (summer 2016) from ENSTA, France, directed by Luc Jaulin.

I have also supervised high-school students, whom I invite, every summer, to be interns in my research lab. Such internships are offered through a program called NEXUS (https://www.utep.edu/-engineering/academic-programs/k12-outreach/nexus.html) from UTEP's College of Engineering. NEXUS is a shadowing program but, in my research lab, we go much beyond a mere shadowing program. Each summer, I have mentored high-school students who spend the summer as interns in my lab, not only learning how to code but also learning about our research and participating in our research via testing. Each student has to pick two projects, in addition to working in research with us and on their own projects, they have to learn how to pitch a project idea and then to report on it weekly. In total, I have mentored about 20 high-school students in summer since 2010.

2.5 Letters peers who have observed classes or reviewed course materials

Supplemented as appendix to this teaching portfolio are letters from peers with whom I have taught or who have been able to observe me while teaching. Supplemented letters are from:

1. Dr. Christina Convertino, an assistant professor of Teacher Education at UTEP, is part of the NSF RED project we have in the department. She and her post-doctoral researcher Angelica Monnarez observed many of my courses since fall 2016. I asked Christina to come to my class once to specifically observe and assess my teaching. Her observations are summarized in the letter available as appendix document.

2.

3.

---

3 Appendix: List of Supporting Documents

3.1 Course information and evaluations

- Students evaluations of my teaching (available on webpage at: http://martineceberio.fr/-miscellaneous/dossier)

3.2 Teaching improvement efforts

- Evidence from Piazza: A sample of course piazza sites
  - CS3360, fall 2012: piazza.com/utep/fall2012/cs3360/home
  - CS3350, fall 2013: piazza.com/utep/fall2013/cs3350/home
  - CS2401, spring 2014: piazza.com/utep/spring2014/cs2401/home
  - CS1401, spring 2016: piazza.com/utep/spring2016/cs14011/home
  - CS1301, fall 2017: piazza.com/utep/fall2017/cs1301/home
  - CS1101, fall 2017: piazza.com/utep/fall2017/cs1101/home
  - CS1190, fall 2017: piazza.com/utep/fall2017/cs1190/home
- Documents used in class:
  - Daily survey
  - Informal surveys/questions during the semester
- Socrative quizzes evidence

3.3 Curriculum development

- New Problem-solving courses:
  1. CS4365/5354: syllabus, exams
  2. CS1190:syllabus, examples of slides, material, exams, students' presentations
- Syllabi
  1. CS2401: spring 2014
  2. CS1401 and CS1301/1101: spring 2015, spring 2016, fall 2017
- Examples of labs developed for CS1401 (zipped files): spring 2015, spring 2016, fall 2017.
- Examples of assignments:
  1. Projects for CS3360
  2. Projects for CS3350

3.4 Letters from peers who have observed classes or reviewed course materials

1. Dr Christina Convertino, Assistant Professor in Teacher Education.

3.5 Letters from research students

1. Leobardo Valera, currently a Ph.D. student about to graduate (MS in Computational Sciences in Dec. 2014)
2. Angel Garcia Contreras, Ph.D. student in Computer Science
3. Cristian Ayub, Undergraduate student in Computer Science
4. Phillip Hassoun, Undergraduate student in Computer Science
5. Sairy Cohen, Undergraduate student in Computer Science