ANALYSING THE EFFECTS OF DATA AUGMENTATION AND HYPER-PARAMETERS FOR TEXT CLASSIFICATION WITH CONVOLUTIONAL NEURAL NETWORKS

JONATHAN K. QUIJAS

Department of Computer Science

APPROVED:

Olac Fuentes, Chair, Ph.D.

Monika Akbaré, Ph.D.

David Novick, Ph.D.

Pablo Arenaz, Ph.D.
Dean of the Graduate School

to my

FAMILY

thanks for everything

ANALYSING THE EFFECTS OF DATA AUGMENTATION AND HYPER-PARAMETERS FOR TEXT CLASSIFICATION WITH CONVOLUTIONAL NEURAL NETWORKS

by

JONATHAN K. QUIJAS

THESIS

Presented to the Faculty of the Graduate School of

The University of Texas at El Paso

in Partial Fulfillment

of the Requirements

for the Degree of

MASTER OF SCIENCE

Department of Computer Science

THE UNIVERSITY OF TEXAS AT EL PASO

May 2017

Acknowledgements

Abstract

Solving systems of linear equations is a common computational problem well known to mathematicians, scientists and engineers. Several algorithms exist for solving this problem. However, when the equations contain *interval coefficients* (i.e., intervals in which the desired coefficient values are known to lie), the problem may not be solvable in any reasonable sense. In fact, it has been shown that the general problem of solving systems of linear equations with interval coefficients is NP-*hard*, i.e., extremely difficult and (it is believed) unsolvable; thus, no feasible algorithm can ever be developed that will solve all particular cases of this problem.

It turns out, though, that the widths of the interval coefficients are quite small in a large number of the linear systems having interval coefficients. This becomes readily apparent when we learn that the intervals typically come from measurements.

Any measurement of a physical quantity is limited by the precision and accuracy of the measuring device. To be of practical use, the measuring devices used in science and industry must be reasonably accurate. This implies that, for the most part, the actual values associated with measurements lie within relatively narrow intervals. Indeed, manufacturers often guarantee the error of their instruments to be very small.

Thus, we desire to look only at *narrow-interval* coefficients when considering the development of an algorithm for solving linear systems with interval coefficients. As there already exists an algorithm that solves most such systems, developing such an algorithm seems indeed promising. Therefore, the goal of this thesis is to answer the following question:

> *Can a feasible algorithm be developed for the general problem of solving systems of linear equations with narrow-interval coefficients?*

We show here that this problem, that of solving systems of linear equations with narrow-

interval coefficients, is NP-hard; thus, we do not consider it possible to develop a feasible algorithm that will solve all particular cases of this problem.

Table of Contents

Chapter 1

Introduction

1.1 Brief Overview of Deep Neural Networks

Deep convolutional neural networks have seen an enormous amount of success on a wide array of application, from scene interpretation to self-driving vehicles and art generation[CITE]. Natural language processing tasks are no exception to the range of problems deep learning can solve, from sentiment analysis to language modeling. In this work, we focus our efforts to studying convolutional and recurrent neural networks for text classification. Especifically, we analyse how well modern neural networks models performed using scientific abstract text data from multiple disciplines such as astro-physics and computer science.

1.2 Training a Neural Network

A neural network is a function $f(\boldsymbol{x};\boldsymbol{\theta})$ that maps its input $\boldsymbol{x}$ to some response variable y. When we *train* a neural network, we *learn* the model parameters, or weights, $\boldsymbol{\theta}$ that minimize some cost function $J(\boldsymbol{\theta})$. For a regression task, where the model's output is a continuous variable, a common cost function is the **Mean Square Error**:

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m}(y_i - f(\boldsymbol{x}_i;\boldsymbol{\theta}))^2$$

For categorical or discrete output variables found in e.g. classification tasks, we use the **Categorical Cross-Entropy**:

$$J(\boldsymbol{\theta}) = -\mathbb{E}_{\boldsymbol{x},y\sim\hat{p}_{data}} \log p(y|\boldsymbol{x};\boldsymbol{\theta})$$

Given a *training* set of observations $\boldsymbol{x}_i$ and their true labels y_i, we compute weights that minimize the cost, or error, via maximum likelihood (ML) estimation:

$$\boldsymbol{\theta}_{ML} = \underset{\boldsymbol{\theta}}{\text{argmax}} \sum_{i}^{m} \log P(y_i|\boldsymbol{x}_i;\boldsymbol{\theta})$$

,

which one can see is the equivalent of computing the weights that ***minimize*** the cross-entropy cost function.

1.3 Bias-Variance Tradeoff

When learning a neural network's weights, we use a *training set* so that we can later generalize previously unseen data with high accuracy (or some other determined metric). That means that during training, we obtain $\boldsymbol{\theta}_{ML}$ by minimizing $J_{train}(\boldsymbol{\theta})$, but we care about having low $J_{test}(\boldsymbol{\theta})$ i.e. low cost on test data points.

Overfitting occurs when a network is able to predict its training set extremely well i.e. very close to zero error, but fails to predict unseen data points. This is because the network's weights have been extremely fine-tuned to *fit* its training data, but do not fit or represent data points outside of its training sample. An overfitted model is said to have large **variance** and small **bias**. Conversely, underfitting occurs when the model fails to predict the training set because it generalizes too harshly. This model is said to have large bias and small variance.

Because of the commonly large amount of weights in deep convolutional networks, it is easy to overfit even a moderate size training set[CITE]. Many techniques exist to avoid overfitting in a neural network [EXPAND]

In this work, we study the effects of multiple regularization techniques used to avoid overfitting in a neural network[REFER TO CHAPTER].

-Describe and cite conv nets for text classification -Describe the conv net pipeline -Add figure of pipeline -Comment that LSTM have made huge progress -Describe difficulty with smaller datasets Convolutional neural networks obtain state of the art results on image and text processing tasks.

1.4 Data Augmentation: Increasing Training Sample Size

-Describe usual augmentation schemes for vision tasks -describe why they work, small changes, same class -propose augmentation scheme and refer to corresponding chapter

Chapter 2

Text Classification with Deep Neural Networks

2.1 Word Embeddings

A very common and simple vector respresentation for words is the one-hot representation. The length of a one-hot vector is the size of the data vocabulary i.e. how many distinct words are found in our data set. For any word, its one-hot vector is zeros everywhere except for a 1 at the word's index. This representation, although simple, fails to capture any meaning other than an identifier. Neural language models i.e. a language model learned using a network, learn to represent words as continuous, dense vectors. These dense, continouos vector representations are commonly called word embeddings. Due to the of the underlying algorithm used to learn these word embeddings, similar words tend to lie closer to each other on embedding space. Because of this, word embeddings are said to capture semantic relations, and thus encode more information than just a word identifier.

[SHOW EMBEDDING PROJECTION]

2.2 Convolutional Neural Networks

Convolutional neural networks are known for their abilities to learn high-level features from raw data. As input signals advance forward through the network, they produce latent signals as linear conbinations with learned parameters, have non-linearities applied to them, and have a pooling or selection mechanism based on simple functions such as the average or maximum operations.

When dealing with image data, images are convolved with multiple filters, each convolution applied to overlapping subimages called as receptive fields. This localized convolution process leads to discovery of low level features of images in the training set such as edges. As data flows forward through the model, higher level features are discovered e.g. wheels or headlights in a vehicle image dataset.

These networks are comprised of *feature maps*. A feature map is a **convolution** layer paired with a **pooling** layer afterwards. The convolution stage creates *activations*, whereas the pooling stage reduces dimensionality and creates translation invariance[CITE].

[INCLUDE IMAGE]

We can take advantage of word embeddings and apply convolutions to text in a fashion akin to convolutions with image data. In a similar manner, we apply convolutions to sub-regions of the input text i.e. bi-grams, tri-grams, etc. This

[DEVELOP]

2.3 Input Representation: Integer Sequences to Word Embeddings

Given a set of texts $\boldsymbol{w_1}, ..., \boldsymbol{w_m}$, we build a vocabulary $\mathbb{V}$ and a bag-of-words model from $\mathbb{V}$ to create a mapping $BoW : \mathbb{V} \mapsto \{1, ..., |\mathbb{V}|\}$. We represent a training text $\boldsymbol{w_i}$ as a sequence of integers, each integer being simply a word index in $\mathbb{V}$. Of course, the training texts will be of variable length. In order to enforce uniform input size for our neural networks, we apply **zero-padding**. For any arbitrary training instance $BoW(\boldsymbol{w}) = \boldsymbol{x} = x_1, ..., x_k$, we enforce that $k = n$, for the specified input size n. Thus, if $k<n$, we transform it into $\boldsymbol{x}_{pad} = x_1, ..., x_k, 0_{k+1}, ..., 0_n$. Conversely, if $k>n$, we simply truncate $\boldsymbol{x}$ to be of size n.

Having converted a text into a sequence of word indexes i.e. integers, we then convert this sequence into an embedding matrix. In order to convert a word into a dense, realvalued vector, we use a token-to-embedding dictionary to map a token to its corresponding embedding form.

Thus, for an input text $\boldsymbol{w}$, we transform it into a sequence of integers $\boldsymbol{x}$, and from there into $\mathbf{E} \in \mathbb{R}^{n \times d}$, where d is the embedding size.

Algorithm 1 Extract a keyword noun phrase from input sentence

```
1: procedure CSO($\{\{np^1_1, ..., np^1_{|np^1|}\}, ..., \{np^n_1, ..., np^n_{|np^n|}\}\}$)
2:     for $k = 1$ to $n$ do
3:         $\boldsymbol{C^k} = \begin{bmatrix} np^k_1 \\ \vdots \\ np^k_{|np_k|} \end{bmatrix} \times \begin{bmatrix} np^k_1 \\ \vdots \\ np^k_{|np_k|} \end{bmatrix}^T$
4:         $c_k = \frac{1}{|np_i|^2} \sum_{i=1}^{|np_i|} \sum_{j=1}^{|np_i|} \boldsymbol{C^k_{(i,j)}}$
5:     end for
       return $\operatorname*{argmin}_k c$
6: end procedure
```

The procedure *CSO* receives as input a set of noun phrases $\mathbf{np^i} = \{np^i_1, ..., np^i_{|\mathbf{np^i}|}\}$ corresponding to the ith sentence. Each noun phrase np^i_1 is comprised of one or more word embedding vectors, corresponding to the noun phrase's tokens mapped to their vector representations. For each noun phrase np^i_1, we compute the average cosine similarity between all the noun phrase's tokens. The algorithm returns the noun phrase which minimizes its average cosine similarity metric. The reasoning behind this noun phrase selection algorithm is as follows. A noun phrase with a very informative token i.e. word embedding will contain a token which will stand out from the rest of the tokens in the noun phrase. We assume that this can be measured by low cosine similarity between an informative token and all its other noun phrase tokens. We therefore compute, for all noun phrases in a sentence, the mean *within-noun-phrase* average cosine similarity, as described in algorithm 1.

Chapter 3

Improving Model Performance via Regularization and Data Augmentation

In this chapter we will introduce the regularization and data augmentation methods we use for our tests.

3.1 Dropout

Dropout is one of the most popular regularization mechanism used in deep networks today. Simply put, dropout *deactivates* a unit with probability p_{drop}. This forces the network to not rely on certain activations, since any unit will only be *active* with probability $1 - p_{drop}$[CITE].

[IMAGE]

3.2 Keyword Extraction: Cosine Similarity Optimization (CSO)

As described in [CITE SECTION], in order to convert a word into a word embedding, we use a token-to-embedding dictionary to map a token to its corresponding embedding form. This dictionary is typically obtained after learning a language model using a neural network.

One property of this word embedding space is that words can now be represented as points in embedding space. Another property and perhaps the most important is that semantic relationships are captured in this embedding space and can be measured between any two words using the cosine similarity metric. This is an attractive property that allows for term comparison by semantic relationships. These relationships can range from complete similarity i.e. the same term, to term orthogonality i.e. unrelated words, and even capture the direction of the relation e.g. negative values of cosine similarity.

The cosine similarity between two vectors can be computed as follows:

$$\frac{\mathbf{x} \cdot \mathbf{y}}{\|\mathbf{x}\|\|\mathbf{y}\|}$$

If we normalize all embeddings to have unit norm, then the cosine similarity between two words is simply a dot product computation.

3.3 Dataset Augmentation: Padding with Extracted Keywords

As described in [CSO SECTION], can use a cosine similarity-based metric to extract keywords from text. Consider a text that yields keyword indexes $WordToIndex(CSO(text)) = w_1, ..., w_j$ and input $\boldsymbol{x}_{pad} = x_1, ..., x_k, 0_{k+1}, ..., 0_n$, we concatenate the non-zero entries $x_1, ..., x_k$ with the indexes of the keywords extracted from the text corresponding to $\boldsymbol{x}_{pad}$.

3.4 Dataset Augmentation: Padding with Similar Keywords

[LSH]

Chapter 4

Model, Dataset, and Final Pipeline Description

In this chapter we describe our model architecture. We start by describing our choice of layers, followed by our network's final architecture. We conclude the chapter by describing our dataset.

4.1 Layer Descriptions

4.1.1 Embedding Layer

This layer maps an word index, or integer, into its corresponding embedding. This is a layer in the neural network because the embeddings can be further fine-tuned during training. Because of the large number of parameters in this layer (number of words allowed times embedding size), we add a L_2 regularization.

4.1.2 Feature Maps: Convolution + Pooling

We refer to a pair of convolutional layer followed by a pooling layer as a feature map[CITE].

4.1.3 Gated Recurrent Unit Layer

4.2 Model Description

The network's general architecture is as follows:

- Input layer: word index vector
- Embedding layer: maps word index vector to embedding matrix
- Dropout layer
- Feature Map 1:
  - Convolution layer: number of kernels:32, activation: rectified linear
  - MaxPooling layer
- Dropout layer
- Feature Map 2:
  - Convolution layer: number of kernels:32, activation: rectified linear
  - MaxPooling layer
- Dropout layer
- Gated Recurrent Unit layer
- Dropout layer
- Dense layer: output size: number of classes, activation: softmax

[IMAGE OF ARCHITECTURE]

4.3 Dataset Description

[DESCRIBE AND CITE EMBEDDINGS]

We gathered scientific paper abstracts from the online repository Arxiv.org. We scraped papers for 5 departments: computer science, mathematics, astrophysics, physics, quantitative biology, and quantititative finance.

4.4 Final Pipeline

[INCLUDE PIPELINE DIAGRAM]

Chapter 5

Solving Narrow-Interval Linear Systems Is NP-Hard: New Result

In this chapter we present the main result upon which this thesis is centered—a new theorem and a proof for it based upon the reduction[1] of a system of (arbitrary) interval linear equations to a system of narrow-interval linear equations.

5.1 Definitions

Definition 1 We say that the (non-zero) number $\tilde{x}$ represents a number x with a *relative accuracy* $\delta > 0$ if $\dfrac{|x-\tilde{x}|}{|\tilde{x}|} \le \delta$.

Definition 2 By *relative half-width* of the interval $\mathbf{x} = [x^-, x^+]$ where $0 \notin [x^-, x^+]$, we mean the smallest number $\delta > 0$ for which every number in $\mathbf{x}$ is represented with a relative accuracy δ by $\tilde{x} = \dfrac{x^- + x^+}{2}$.

Proposition It can be seen that the relative half-width of $\mathbf{x}$ where $0 \notin [x^-, x^+]$ is

$$\max_{x \in [x^-, x^+]} \frac{|x-\tilde{x}|}{|\tilde{x}|} = \frac{x^+ - x^-}{2|\tilde{x}|}.$$

Definition 3 By *relative width* W^{rel} of an interval $\mathbf{x} = [x^-, x^+]$ where $0 \notin [x^-, x^+]$, we mean twice the relative half-width of $\mathbf{x}$, i.e.,

$$W^{rel}([x^-, x^+]) = \frac{x^+ - x^-}{|\tilde{x}|}.$$

[1]The reduction used is called a *polynomial-time one-one reduction*, a special case of polynomial-time many-one reductions.

Definition 4 We say that an interval $\mathbf{x}$ is *δ-narrow in the sense of relative accuracy* if $W^{rel}(\mathbf{x}) \leq \delta$.

Definition 5 We say that an interval $\mathbf{x}$ is *δ-narrow* if it is both δ-narrow in the sense of absolute accuracy and δ-narrow in the sense of relative accuracy.

5.2 Theorem

Theorem 1 *For every $\delta > 0$, the problem of solving systems of interval linear equations with δ-narrow intervals is computationally intractable (*NP*-hard).*

Corollary 1 *For every $\Delta > 0$, the problem of solving systems of interval linear equations with intervals Δ-narrow in the sense of absolute accuracy is computationally intractable (*NP*-hard).*

Corollary 2 *For every $\delta > 0$, the problem of solving systems of interval linear equations with intervals δ-narrow in the sense of relative accuracy is computationally intractable (*NP*-hard).*

5.3 Proof

5.3.1 Part I: Reduction of Interval Linear System to Narrow-Interval Linear System

We have seen that the general problem of solving interval linear equation systems (**??**) is NP-hard. We now intend to show that this general problem can be reduced to the problem of solving a system of interval linear equations with δ-narrow intervals.

Let us start with an arbitrary interval linear equation system (**??**). To reduce it to a δ-narrow interval linear equation system, we introduce new variables w_i, y_{ij} and z_{ij} for all $i = 1, \ldots, m$ and $j = 1, \ldots, n$. For the enlarged list of $m + n + 2mn$ variables (i.e., x_j, w_i, y_{ij} and z_{ij}) we introduce the following new system of $m+n+2mn$ δ-narrow interval linear equations[2]:

$$\sum_{j=1}^{n} y_{ij} + \sum_{j=1}^{n}[1-\frac{\delta}{2}, 1+\frac{\delta}{2}]z_{ij} + [1-\frac{\delta}{2}, 1+\frac{\delta}{2}]w_i = c_i, \tag{5.1}$$

$$w_i = \gamma_i, \tag{5.2}$$

$$y_{ij} - \mu_{ij}x_j = 0, \tag{5.3}$$

$$z_{ij} - \nu_{ij}x_j = 0, \tag{5.4}$$

where $\delta > 0$ and the numerical (non-interval) coefficients c_i, $\gamma_i \geq 0$, μ_{ij} and $\nu_{ij} \geq 0$ will be chosen in such a way that for every solution of this δ-narrow interval linear equation system (5.1)–(5.4) we have

$$c_i - [1-\frac{\delta}{2}, 1+\frac{\delta}{2}]w_i = [b_i^-, b_i^+] \tag{5.5}$$

and

$$y_{ij} + [1-\frac{\delta}{2}, 1+\frac{\delta}{2}]z_{ij} = [a_{ij}^-, a_{ij}^+]x_j \tag{5.6}$$

for all $i = 1, \ldots, m$ and $j = 1, \ldots, n$.

[2]For clarity, non-interval coefficients can be thought of as intervals of zero width; e.g., the coefficient for each variable y_{ij} is $[1, 1]$ and each coefficient c_i is equivalent to the interval $[c_i, c_i]$.

Let us first find the values for c_i and γ_i. From equation (5.2) we know that $w_i = \gamma_i$. By substituting this expression into equation (5.5) we get

$$c_i - [1 - \frac{\delta}{2}, 1 + \frac{\delta}{2}]\gamma_i = [b_i^-, b_i^+].$$

Intervals are equal *if and only if* their endpoints coincide. Since we are looking for a solution with $\gamma_i \geq 0$, the equations for the endpoints take the following forms:

$$c_i - (1 - \frac{\delta}{2})\gamma_i = b_i^+ \tag{5.7}$$

and

$$c_i - (1 + \frac{\delta}{2})\gamma_i = b_i^-. \tag{5.8}$$

Subtracting the second equation from the first we get

$$\delta \cdot \gamma_i = b_i^+ - b_i^-,$$

and hence,

$$\boxed{\gamma_i = \frac{1}{\delta}(b_i^+ - b_i^-).} \tag{5.9}$$

Substituting this value into equation (5.7), we get

$$c_i - (1 - \frac{\delta}{2})\frac{1}{\delta}(b_i^+ - b_i^-) = b_i^+$$

from which we get

$$c_i - (\frac{1}{\delta} - \frac{1}{2})(b_i^+ - b_i^-) = b_i^+,$$

and hence,

$$\boxed{c_i = \frac{1}{2}(b_i^+ + b_i^-) + \frac{1}{\delta}(b_i^+ - b_i^-).} \tag{5.10}$$

Now let us find the values for μ_{ij} and ν_{ij}. From equations (5.3) and (5.4) we conclude that $y_{ij} = \mu_{ij}x_j$ and $z_{ij} = \nu_{ij}x_j$. Substituting these expressions into equation (5.6) we get

$$\mu_{ij}x_j + [1 - \frac{\delta}{2}, 1 + \frac{\delta}{2}]\nu_{ij}x_j = [a_{ij}^-, a_{ij}^+]x_j.$$

For this equality to be true for all x_j, coefficients for x_j on both sides of the equation must coincide. Thus, we conclude that

$$\mu_{ij} + [1 - \frac{\delta}{2}, 1 + \frac{\delta}{2}]\nu_{ij} = [a_{ij}^-, a_{ij}^+].$$

As stated before, for intervals to be equal their endpoints must coincide, and since we are looking for a solution for $\nu_{ij} \geq 0$, the equations for the endpoints take the following forms:

$$\mu_{ij} + (1 + \frac{\delta}{2})\nu_{ij} = a_{ij}^+ \tag{5.11}$$

and

$$\mu_{ij} + (1 - \frac{\delta}{2})\nu_{ij} = a_{ij}^-. \tag{5.12}$$

Subtracting the second equation from the first we get

$$\delta \cdot \nu_{ij} = a_{ij}^+ - a_{ij}^-,$$

and hence,

$$\boxed{\nu_{ij} = \frac{1}{\delta}(a_{ij}^+ - a_{ij}^-).} \tag{5.13}$$

Substituting this value back into equation (5.11) we get

$$\mu_{ij} + (1 + \frac{\delta}{2})\frac{1}{\delta}(a_{ij}^+ - a_{ij}^-) = a_{ij}^+$$

which leads to

$$\mu_{ij} + (\frac{1}{\delta} + \frac{1}{2})(a_{ij}^+ - a_{ij}^-) = a_{ij}^+,$$

and hence,

$$\boxed{\mu_{ij} = \frac{1}{2}(a_{ij}^+ + a_{ij}^-) - \frac{1}{\delta}(a_{ij}^+ - a_{ij}^-).} \tag{5.14}$$

5.3.2 Part II: The Two Systems Are "Equivalent"

Let us show that every system (**??**) of interval linear equations is "equivalent" to the δ-narrow interval linear equation system (5.1)–(5.4) in the following sense:

Claim 1 If $(x_1, \ldots, x_n)$ is a solution of the original system (**??**), then for the same values $x_1, \ldots, x_n$, and for some values $w_1, \ldots, w_m$, $y_{11}, \ldots, y_{mn}$ and $z_{11}, \ldots, z_{mn}$, the extended tuple

$$(x_1, \ldots, x_n, w_1, \ldots, w_m, y_{11}, \ldots, y_{mn}, z_{11}, \ldots, z_{mn}) \tag{5.15}$$

is a solution of the δ-narrow interval system (5.1)–(5.4).

Claim 2 Conversely, if tuple (5.15) is a solution of the δ-narrow interval linear equation system (5.1)–(5.4), then $(x_1, \ldots, x_n)$ is a solution of the original system (**??**) for the same values $x_1, \ldots, x_n$.

We will now prove that the two systems are indeed "equivalent" in the above sense.

Proof of Claim 1

Let us assume that $(x_1, \ldots, x_n)$ is a solution of the original system (**??**). By definition, if $(x_1, \ldots, x_n)$ is a solution to the original system (**??**), then there must exist values $a_{ij} \in [a_{ij}^-, a_{ij}^+]$ and $b_i \in [b_i^-, b_i^+]$ such that

$$\sum_{j=1}^{n} a_{ij} x_j = b_i \tag{5.16}$$

for all $i = 1, \ldots, m$. With this in mind, let us introduce new parameters $\alpha_{ij} = a_{ij} - a_{ij}^-$ and $\beta_i = b_i - b_i^-$; thus, $a_{ij} = a_{ij}^- + \alpha_{ij}$ and $b_i = b_i^- + \beta_i$. Substituting these expressions into (5.16) we get

$$\sum_{j=1}^{n} (a_{ij}^- + \alpha_{ij}) x_j = b_i^- + \beta_i. \tag{5.17}$$

Since $a_{ij} \in [a_{ij}^-, a_{ij}^+]$, we conclude that $\alpha_{ij} \in [0, a_{ij}^+ - a_{ij}^-]$. Similarly, since $b_i \in [b_i^-, b_i^+]$, we conclude that $\beta_i \in [0, b_i^+ - b_i^-]$.

Now let us show that there exists an extended tuple (5.15) that is a solution to system (5.1)–(5.4) for the same values $x_1, \ldots, x_n$ and for appropriately chosen values of $w_1, \ldots, w_m$, $y_{11}, \ldots, y_{mn}$ and $z_{11}, \ldots, z_{mn}$. In order to have equations (5.2)–(5.4) automatically satisfied, we will take $w_i = \gamma_i$, $y_{ij} = \mu_{ij} x_j$ and $z_{ij} = \nu_{ij} x_j$. It is thus sufficient to show that equation (5.1) is satisfied for all $i = 1, \ldots, m$; in other words, we need to show that there exist values $\kappa_i \in [1 - \frac{\delta}{2}, 1 + \frac{\delta}{2}]$ and $\lambda_{ij} \in [1 - \frac{\delta}{2}, 1 + \frac{\delta}{2}]$ such that

$$\sum_{j=1}^{n} y_{ij} + \sum_{j=1}^{n} \lambda_{ij} z_{ij} + \kappa_i w_i = c_i \tag{5.18}$$

for all $i = 1, \ldots, m$.

Note that for any given i, if $\gamma_i = 0$ (and thus $w_i = 0$), then $\kappa_i w_i = 0$, implying that κ_i can be *any value* as long as $\kappa_i \in [1 - \frac{\delta}{2}, 1 + \frac{\delta}{2}]$ (e.g., $\kappa_i = 1$). Likewise, for any given i and j, if $\nu_{ij} = 0$ (and thus $z_{ij} = 0$), then $\lambda_{ij} z_{ij} = 0$, implying that λ_{ij} can be *any value* as long as $\lambda_{ij} \in [1 - \frac{\delta}{2}, 1 + \frac{\delta}{2}]$ (e.g., $\lambda_{ij} = 1$). Therefore, for the remainder of the proof we will be choosing κ_i only for those i for which $\gamma_i > 0$ (i.e., $b_i^- < b_i^+$), and we will be choosing λ_{ij} only for those i and j for which $\nu_{ij} > 0$ (i.e., $a_{ij}^- < a_{ij}^+$).

We have chosen w_i and c_i for which the equation (5.5) is true, i.e., for which

$$c_i - [1 - \frac{\delta}{2}, 1 + \frac{\delta}{2}] w_i = [b_i^-, b_i^+].$$

Crudely speaking, this equality means that when we take different values for κ_i such that $\kappa_i \in [1 - \frac{\delta}{2}, 1 + \frac{\delta}{2}]$, the set

$$c_i - [1 - \frac{\delta}{2}, 1 + \frac{\delta}{2}] w_i$$

of possible values of $c_i - \kappa_i w_i$ coincides with the interval $[b_i^-, b_i^+]$. In particular, since $b_i \in [b_i^-, b_i^+]$, there must exist a value $\kappa_i \in [1 - \frac{\delta}{2}, 1 + \frac{\delta}{2}]$ for which

$$c_i - \kappa_i w_i = b_i = b_i^- + \beta_i.$$

From this equation we can find the exact value of κ_i.

We know from equation (5.2) that $w_i = \gamma_i$, so by substitution we get

$$c_i - \kappa_i \gamma_i = b_i^- + \beta_i,$$

and hence,

$$\kappa_i = \frac{1}{\gamma_i}(c_i - b_i^- - \beta_i).$$

Substituting the values of γ_i and c_i from equations (5.9) and (5.10) we get

$$\kappa_i = \frac{\delta}{b_i^+ - b_i^-}(\frac{1}{2}(b_i^+ + b_i^-) + \frac{1}{\delta}(b_i^+ - b_i^-) - b_i^- - \beta_i).$$

Simplifying, we get

$$\kappa_i = 1 + \frac{\delta}{b_i^+ - b_i^-}(\frac{1}{2}(b_i^+ + b_i^-) - b_i^- - \beta_i)$$

from which we get

$$\kappa_i = 1 + \frac{\delta}{b_i^+ - b_i^-}(\frac{1}{2}(b_i^+ - b_i^-) - \beta_i),$$

and hence,

$$\boxed{\kappa_i = 1 + \frac{\delta}{2} - \frac{\delta \cdot \beta_i}{b_i^+ - b_i^-}.} \tag{5.19}$$

We have also chosen y_{ij} and z_{ij} for which the equation (5.6) is true, i.e., for which

$$y_{ij} + [1 - \frac{\delta}{2}, 1 + \frac{\delta}{2}]z_{ij} = [a_{ij}^-, a_{ij}^+]x_j$$

Crudely speaking, this equality means that when we take different values for λ_{ij} such that $\lambda_{ij} \in [1 - \frac{\delta}{2}, 1 + \frac{\delta}{2}]$, the set

$$y_{ij} + [1 - \frac{\delta}{2}, 1 + \frac{\delta}{2}]z_{ij}$$

of possible values of $y_{ij} + \lambda_{ij}z_{ij}$ coincides with the interval $[a_{ij}^-, a_{ij}^+]x_j$. In particular, since $a_{ij} \in [a_{ij}^-, a_{ij}^+]$, there must exist a value $\lambda_{ij} \in [1 - \frac{\delta}{2}, 1 + \frac{\delta}{2}]$ for which

$$y_{ij} + \lambda_{ij}z_{ij} = (a_{ij}^- + \alpha_{ij})x_j.$$

From this equation we can find the exact value of λ_{ij}.

We conclude from equations (5.3) and (5.4) that $y_{ij} = \mu_{ij}x_j$ and $z_{ij} = \nu_{ij}x_j$, so by substitution we get

$$\mu_{ij}x_j + \lambda_{ij}\nu_{ij}x_j = (a_{ij}^- + \alpha_{ij})x_j,$$

and hence,

$$\lambda_{ij} = \frac{1}{\nu_{ij}}(a_{ij}^- + \alpha_{ij} - \mu_{ij}).$$

Substituting the values of the coefficients ν_{ij} and μ_{ij} from equations (5.13) and (5.14) we get

$$\lambda_{ij} = \frac{\delta}{a_{ij}^+ - a_{ij}^-}(a_{ij}^- + \alpha_{ij} - \frac{1}{2}(a_{ij}^+ + a_{ij}^-) + \frac{1}{\delta}(a_{ij}^+ - a_{ij}^-)).$$

Simplifying, we get

$$\lambda_{ij} = 1 + \frac{\delta}{a_{ij}^+ - a_{ij}^-}(a_{ij}^- + \alpha_{ij} - \frac{1}{2}(a_{ij}^+ + a_{ij}^-))$$

from which we get

$$\lambda_{ij} = 1 - \frac{\delta}{a_{ij}^+ - a_{ij}^-}(\frac{1}{2}(a_{ij}^+ - a_{ij}^-) - \alpha_{ij}),$$

and hence,

$$\boxed{\lambda_{ij} = 1 - \frac{\delta}{2} + \frac{\delta \cdot \alpha_{ij}}{a_{ij}^+ - a_{ij}^-}.} \tag{5.20}$$

To show that these values for κ_i and λ_{ij} are indeed the ones desired, we first note that, since $\beta_i \in [0, b_i^+ - b_i^-]$ and $\alpha_{ij} \in [0, a_{ij}^+ - a_{ij}^-]$, we can conclude that

$$\frac{\beta_i}{b_i^+ - b_i^-} \in [0, 1]$$

and

$$\frac{\alpha_{ij}}{a_{ij}^+ - a_{ij}^-} \in [0, 1].$$

Thus,

$$\frac{\delta \cdot \beta_i}{b_i^+ - b_i^-} \in [0, \delta]$$

and

$$\frac{\delta \cdot \alpha_{ij}}{a_{ij}^+ - a_{ij}^-} \in [0, \delta]$$

from which it follows that

$$1 + \frac{\delta}{2} - \frac{\delta \cdot \beta_i}{b_i^+ - b_i^-} \in [1 - \frac{\delta}{2}, 1 + \frac{\delta}{2}]$$

and

$$1 - \frac{\delta}{2} + \frac{\delta \cdot \alpha_{ij}}{a_{ij}^{+} - a_{ij}^{-}} \in [1 - \frac{\delta}{2}, 1 + \frac{\delta}{2}],$$

and hence,

$$\kappa_i \in [1 - \frac{\delta}{2}, 1 + \frac{\delta}{2}]$$

and

$$\lambda_{ij} \in [1 - \frac{\delta}{2}, 1 + \frac{\delta}{2}].$$

Our goal is to prove that equation (5.1) is satisfied. By showing that we have found appropriate values for κ_i and λ_{ij} such that equation (5.18) is satisfied, we can do just that. We have already chosen the values for y_{ij} and z_{ij}. Substituting these values into equation (5.18) we get an equivalent equation

$$\sum_{j=1}^{n} \mu_{ij} x_j + \sum_{j=1}^{n} \lambda_{ij} \nu_{ij} x_j + \kappa_i w_i = c_i.$$

This equation, in its turn, is equivalent to the equation

$$\sum_{j=1}^{n} (\mu_{ij} + \lambda_{ij} \nu_{ij}) x_j = c_i - \kappa_i w_i. \tag{5.21}$$

Let us now show that this equation is satisfied and thus prove that the equivalent equation (5.18) is satisfied as well.

- According to our choice of κ_i, the right-hand side of equation (5.21) is equal to b_i.
- According to our choice of λ_{ij}, we have

  $$\mu_{ij} + \lambda_{ij} \nu_{ij} = a_{ij},$$

  and hence, the left-hand side of equation (5.21) is equal to

  $$\sum_{j=1}^{n} a_{ij} x_j.$$

We started with $x_1, \ldots, x_n$ for which $\sum\limits_{j=1}^{n} a_{ij}x_j = b_i$; hence, the left-hand side and the right-hand side of equation (5.21) coincide. Thus, we have proven that equation (5.18) is satisfied.

Since we are able to show that there do indeed exist values $\kappa_i \in [1 - \frac{\delta}{2}, 1 + \frac{\delta}{2}]$ and $\lambda_{ij} \in [1 - \frac{\delta}{2}, 1 + \frac{\delta}{2}]$ such that equation (5.18) is satisfied for all $i = 1, \ldots, m$, it follows that if $(x_1, \ldots, x_n)$ is a solution of system (**??**), then for the same values $x_1, \ldots, x_n$ and for some values $w_1, \ldots, w_m$, $y_{11}, \ldots, y_{mn}$ and $z_{11}, \ldots, z_{mn}$, there exists an extended tuple (5.15) that is a solution of system (5.1)–(5.4). This proves Claim 1.

Proof of Claim 2

Now we must show that the converse is true, namely, that if tuple (5.15) is a solution of the δ-narrow interval linear equation system (5.1)–(5.4), then $(x_1, \ldots, x_n)$ is a solution of the interval linear equation system (**??**) for the same values $x_1, \ldots, x_n$.

Let us assume tuple (5.15) is a solution of system (5.1)–(5.4). For this to be true, equations (5.2)–(5.4) must be satisfied, and thus we can conclude that $w_i = \gamma_i$, $y_{ij} = \mu_{ij}x_j$ and $z_{ij} = \nu_{ij}x_j$. The fact that equation (5.1) is satisfied by a given tuple means that

$$\sum_{j=1}^{n} y_{ij} + \sum_{j=1}^{n} \lambda_{ij}z_{ij} + \kappa_i w_i = c_i \tag{5.22}$$

for some $\lambda_{ij} \in [1 - \frac{\delta}{2}, 1 + \frac{\delta}{2}]$ and $\kappa_i \in [1 - \frac{\delta}{2}, 1 + \frac{\delta}{2}]$. Let us show that

$$\sum_{j=1}^{n} a_{ij}x_j = b_i \tag{5.23}$$

for some values $a_{ij} \in [a_{ij}^-, a_{ij}^+]$ and $b_i \in [b_i^-, b_i^+]$.

From equation (5.22) we conclude that

$$\sum_{j=1}^{n} y_{ij} + \sum_{j=1}^{n} \lambda_{ij}z_{ij} = c_i - \kappa_i w_i.$$

Since $w_i = \gamma_i$, $y_{ij} = \mu_{ij}x_j$ and $z_{ij} = \nu_{ij}x_j$, by substitution we conclude that

$$\sum_{j=1}^{n} (\mu_{ij} + \lambda_{ij}\nu_{ij})x_j = c_i - \kappa_i\gamma_i.$$

Note that this equation has the desired form (5.23) where

$$a_{ij} = \mu_{ij} + \lambda_{ij}\nu_{ij} \tag{5.24}$$

and

$$b_i = c_i - \kappa_i\gamma_i. \tag{5.25}$$

Thus, to complete the proof, we need only show that $a_{ij} \in [a_{ij}^-, a_{ij}^+]$ and $b_i \in [b_i^-, b_i^+]$.

Let us first show that $b_i \in [b_i^-, b_i^+]$. Since $\kappa_i \in [1 - \frac{\delta}{2}, 1 + \frac{\delta}{2}]$, we have that

$$1 + \frac{\delta}{2} \geq \kappa_i \geq 1 - \frac{\delta}{2}.$$

By multiplying all sides of the inequality by $\gamma_i \geq 0$, we conclude that

$$(1 + \frac{\delta}{2})\gamma_i \geq \kappa_i\gamma_i \geq (1 - \frac{\delta}{2})\gamma_i.$$

By subtracting all parts of this inequality from c_i, we conclude that

$$c_i - (1 + \frac{\delta}{2})\gamma_i \leq c_i - \kappa_i\gamma_i \leq c_i - (1 - \frac{\delta}{2})\gamma_i.$$

According to our choice of c_i and γ_i (which led to equations (5.7) and (5.8)), this is equivalent to

$$b_i^- \leq c_i - \kappa_i\gamma_i \leq b_i^+.$$

Thus, the value $b_i = c_i - \kappa_i\gamma_i$ we have chosen indeed belongs to the interval $[b_i^-, b_i^+]$.

Let us now show, in like fashion, that $a_{ij} \in [a_{ij}^-, a_{ij}^+]$. Since $\lambda_{ij} \in [1 - \frac{\delta}{2}, 1 + \frac{\delta}{2}]$, we have that

$$1 - \frac{\delta}{2} \leq \lambda_{ij} \leq 1 + \frac{\delta}{2}.$$

By multiplying all sides of the inequality by $\nu_{ij} \geq 0$, we conclude that

$$(1 - \frac{\delta}{2})\nu_{ij} \leq \lambda_{ij}\nu_{ij} \leq (1 + \frac{\delta}{2})\nu_{ij}.$$

By adding all parts of this inequality to μ_{ij}, we conclude that

$$\mu_{ij} + (1 - \frac{\delta}{2})\nu_{ij} \leq \mu_{ij} + \lambda_{ij}\nu_{ij} \leq \mu_{ij} + (1 + \frac{\delta}{2})\nu_{ij}.$$

According to our choice of μ_{ij} and ν_{ij} (which led to equations (5.11) and (5.12)), this is equivalent to

$$a_{ij}^- \leq \mu_{ij} - \lambda_{ij}\nu_{ij} \leq a_{ij}^+,$$

i.e., the value $a_{ij} = \mu_{ij} - \lambda_{ij}\nu_{ij}$ that we have chosen indeed belongs to the interval $[a_{ij}^-, a_{ij}^+]$.

Thus, we have shown that if tuple (5.15) is a solution of system (5.1)–(5.4), then $(x_1, \ldots, x_n)$ is a solution of system (**??**) for the same values $x_1, \ldots, x_n$. This proves Claim 2.

5.3.3 Conclusion

We saw in part II that the two systems, namely, the interval linear equation system (**??**) and the δ-narrow interval linear equation system (5.1)–(5.4), are "equivalent" in the sense described therein. In part I we saw that the number of equations and unknowns of system (5.1)–(5.4) is bounded by a polynomial of the number of equations and unknowns of system (**??**). Further, the number of steps required for reducing an interval linear equation system (**??**) to a δ-narrow interval linear equation system (5.1)–(5.4) is bounded by a polynomial of the number of equations and unknowns of system (**??**). Thus, it follows from parts I and II that if we have an algorithm $\mathcal{U}$ that can solve δ-narrow interval linear equation systems in polynomial time, then we can solve an arbitrary system (**??**) of interval linear equations in polynomial time by applying this hypothetical algorithm $\mathcal{U}$ to the "equivalent" δ-narrow interval linear equation system (5.1)–(5.4). But, as stated in part I, solving interval linear equation systems is an NP-hard problem.

So, if we can (in general) solve interval linear equation systems in polynomial time, we can solve any problem from the class NP in polynomial time. Therefore, if we can (in general) solve δ-narrow interval linear equation systems in polynomial time, we can solve any problem from the class NP in polynomial time. Thus, we conclude that *the problem of solving δ-narrow interval linear equation systems is* NP*-hard.* Q.E.D.

Chapter 6

Concluding Remarks

6.1 Significance of the Result

Now that we have shown that we cannot solve narrow-interval linear equation systems in general, what does this mean to the computing and mathematical communities? Unless P=NP, attempts at developing a feasible algorithm to solve this problem in general will assuredly fail; however, the very fact that there exists an algorithm for solving a large number of these systems (see [9]) shows that work on solving a subclass of the class of all narrow-interval linear equation systems is certainly a worthwhile endeavor.

6.2 Future Work

The problem now shifts to identifying new subclasses of the class of all narrow-interval linear equation systems for which the problem of solving them is possible with the development of new algorithms. Also, if the general problem (or any problem in the class NP) shows up often enough in industry, science, research, etc., work on improving existing and/or creating new approximation methods (including heuristic and/or statistical methods, where applicable) is certainly warranted. Since we cannot compute the exact bounds for the general case, good approximation methodologies are the most we can hope for or expect.

References

[1] S. Cook, "The complexity of theorem-proving procedures," *Proceedings of the 3rd ACM Symposium on Theory of Computing*, Shaker Heights, Ohio, 1971, pp. 151–158.

[2] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of* NP-*Completeness*, W. H. Freeman, San Francisco, 1979.

[3] R. Karp, "Reducibility among combinatorial problems," in: R. Miller and J. Thatcher (eds.), *Complexity of Computer Computations*, Plenum Press, New York, 1972, pp. 85–103.

[4] R. B. Kearfott and V. Kreinovich (eds.), *Applications of Interval Computations*, Kluwer Academic Publishers, Norwell, MA, 1996.

[5] V. Kreinovich, A. V. Lakeyev and S. I. Noskov, "Optimal solution of interval linear systems is intractable (NP-hard)," *Interval Computations*, 1993, No. 1, pp. 6–14.

[6] V. Kreinovich, A. V. Lakeyev and J. Rohn , "Computational complexity of interval algebraic problems: some are feasible and some are computationally intractable: a survey," in: G. Alefeld and A. Frommer (eds.), *Scientific Computing and Validated Numerics*, Akademie-Verlag, Berlin, 1996, pp. 293–306.

[7] V. Kreinovich, A. V. Lakeyev, J. Rohn and P. Kahl, *Feasible? Intractable? On Computational Complexity of Data Processing and Interval Computations*, Kluwer Academic Publishers, Norwell, MA, 1996 (to appear).

[8] U. Kulisch and W. L. Miranker, *Computer Arithmetic in Theory and Practice*, Academic Press, NY, 1981.

[9] A. V. Lakeyev and V. Kreinovich, "If input intervals are small enough, then interval computations are almost always easy," *Reliable Computing*, Supplement (Extended

Abstracts of APIC'95: International Workshop on Applications of Interval Computations), 1995, pp. 134–139.

[10] L. Levin, "Universal sequential search problems," *Problems of Information Transmission*, 1973, Vol. 9, No. 3, pp. 265–266.

[11] R. E. Moore, "Automatic error analysis in digital computation," *Technical Report LMSD-48421*, Lockheed Missiles and Space Co., Palo Alto, CA, January 1959.

[12] R. E. Moore, *Interval Analysis*, Prentice Hall, Englewood Cliffs, NJ, 1966.

[13] A. Neumaier, *Interval Methods for Systems of Equations*, Cambridge University Press, Cambridge, 1990.

[14] S. G. Rabinovich, *Measurement Errors: Theory and Practice*, American Institute of Physics, NY, 1995.

Curriculum Vitae

Patrick Thor Kahl was born on July 12, 1961. The first son of Ulf Thor Gustav Kahl and Carolyn Kahl, he graduated from Coronado High School, El Paso, Texas, in the spring of 1979. He entered Auburn University in the fall of 1979, and, in the spring of 1982, The University of Texas at El Paso. In 1985 he joined the United States Navy where he served for eight years, most of it aboard the submarine USS Narwhal (SSN671). In the fall of 1993, after being honorably discharged from the navy, Patrick resumed his studies at The University of Texas at El Paso. While pursuing his bachelor's degree in Computer Science he worked as a Teaching Assistant, and as a programmer at the National Solar Observatory at Sunspot, New Mexico. He received his bachelor's degree in Computer Science in the summer of 1994.

In the fall of 1994, he entered the Graduate School of The University of Texas at El Paso. While pursuing a master's degree in Computer Science he worked as a Teaching and Research Assistant, and as the Laboratory Instructor for the 1995 Real-Time Programming Seminar at the University of Puerto Rico, Mayagüez Campus. He was a member of the Knowledge Representation Group and the Rio Grande Chapter of the Association for Computing Machinery.

Permanent address: 6216 Sylvania Way
El Paso, Texas 79912-4927