Anglo-Chinese School (Independent)
Year 4 (2024) Core Mathematics

Mathematical Exploration
Cover Page

Exploration Title: Title
Date of Submission: 22nd of March, 2024

Assessment Criteria

Criteria	Description	Comments	Max	Marks
A	Presentation		4	
B	Mathematical Communication		4	
C	Personal Engagement		3	
D	Reflection		3	
E	Use of Mathematics		6	
		Total	20	

Declaration by Students

We confirm that this is our own work and that we have both worked collaboratively on it. We have acknowledged every source of information used, whether written, oral or visual. We are aware that if we incorporate material from other works or paraphrase without acknowledgement, it is treated as plagiarism which will result in severe penalties.

Janav Nagapatla (07), 4.10 Samuel

Janav N.

Lim Yong Cen Aiden (17), 4.10 Samuel,

Aiden

Signed on the 21st of March, 2024

Contents Page

1 *Introduction*

1.1 What are neural networks?
1.2 Applications of neural networks
1.3 What we aim to do

2 *Structure of a Neural Network*

2.1 Sigmoid neurone
2.2 Full network
2.3 Forward pass

3 *Training a Neural Network by Calculus*

3.1 What happens when we initialise the network?
3.2 How do we rectify the poor performance of the network?

3.2.1 Discovering how poorly the network has performed
3.2.2 How do we change the output to make it more accurate?
3.2.3 Applying Chain Rule to differentiate the sigmoid neurone
3.2.4 Applying Stochastic Gradient Descent

4 *Implementing the Algorithm*

5 *Demonstration of Efficacy*

6 *Conclusion*

7 *References*

A *Appendix of Programmatical Representations*

1 - Introduction

1.1 What are neural networks?

The purpose of a neural network is to tackle the problem of advanced pattern recognition in a nonlinear fashion, like the human brain does. The specific details of neural networks are expounded on later on. (Chollet, 2017)

Neural networks empower computers to engage in pseudo-human cognition. A task of recognising objects in an image is easy to our brains due to millions of years of evolution. However, computers are only capable of arithmetic and therefore, neural networks model the human brain mathematically. (Sanderson, 2017)

1.2 Applications of neural networks

Neural networks find use in many applications, including, but not limited to: Image classification, Natural language processing, and Predictive analytics.

1.3 What we aim to do

We aim to produce and train a neural network that classifies images from the MNIST dataset (LeCun et al., 1998) of handwritten digits as an integer from 0 to 9.

We will be training the neural network using Stochastic Gradient Descent which is a technique relying on linear algebra and calculus to update the network such that it can predict images accurately.

The final product will be a computer program that can detect the integer value of the digit shown and output it in computer-compatible representation.

2 – Structure of a Neural Network

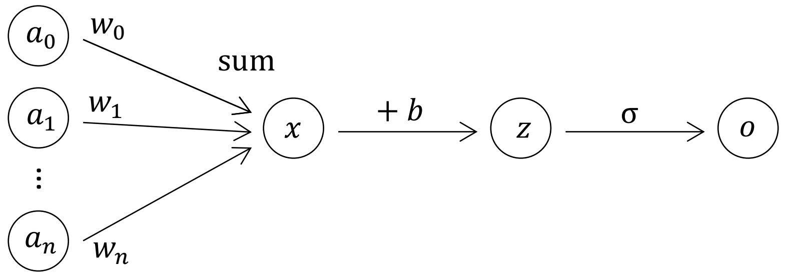
2.1 Sigmoid neurone

Before describing a sigmoid neurone, let's define the logistic sigmoid function. It is: (Lin, 2019)

$$\sigma(x) = \frac{1}{1+e^{-x}}$$

For all real values of x, $\sigma(x)$ is always a value between 0 and 1, as seen from its graph.

A sigmoid neurone can be broken into four parts: the inputs, the weight multiplication, the bias addition, and the sigmoid function (in notation as σ).



Graphics prepared in MS Word by ourselves

In traditional mathematical notation, the sigmoid neurone can be represented as:

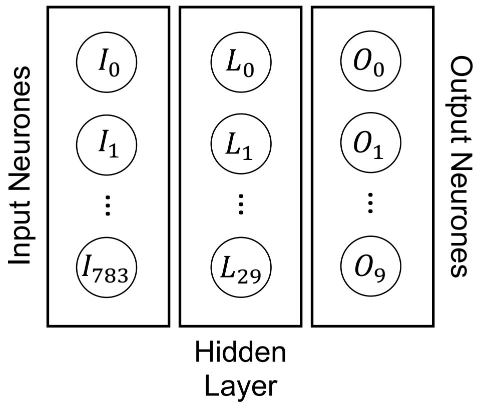
$$o = \sigma\left(b + \sum_{i=0}^{n} a_i w_i\right)$$

2.2 Full network

A fully-connected neural network is simply a series of sigmoid neurones arranged together. We will denote one full sigmoid neurone as one node, as opposed to the 3 nodes in the previous graphical representation. (Sanderson, 2017)

The MNIST dataset's images are 28 pixels wide and tall. Therefore, there are 28^2 (784) input pixel values to the network.

We have 10 possible output values, and therefore, there are 10 output neurones.



The input neurones can be taken straight from the MNIST dataset, and should range from 0 to 1.

Every single node in a connected layer is connected to every single node in the previous layer as a sigmoid layer. For our purposes, we will only be using one hidden layer with 30 neurones (Nielsen, 2015), though in actuality, more can be used with as many neurones as wanted.

There are ten output neurones, each also sigmoid neurones connected to every neurone in the penultimate layer.

2.3 Forward pass

Now that this network is created, how do we predict the integer value of an image? This can be done by first setting the layer I to a matrix of brightness values of the image's pixels.

Then, we can pass each value of I to the first sigmoid neurone of layer L. One may wonder, a sigmoid neurone should have weights and a bias. However, we have not trained the network yet, so we will start by using random weights and biases.

We do likewise for every sigmoid neurone in layer L. Once all values of layer L have been calculated, we will pass each value of layer L into every sigmoid neurone in layer O.

The end result is a matrix of size ten. [0.253, 0.786, 0.998, 0.001 … 0.122, 0.545, 0.876, 0.623]

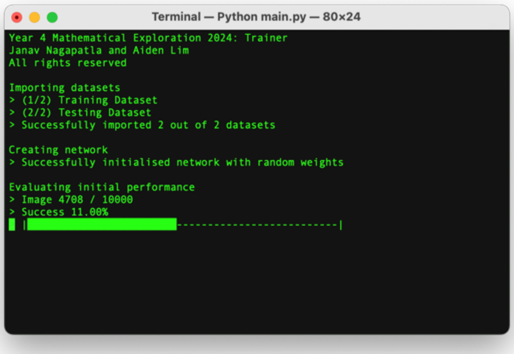
The maximum value is 0.998 and it is at the 3rd position, so the answer is 2 (0, 1, 2 … 9).

This code implementation can be seen in the GitHub repository or appendix.

3 – Training a Neural Network by Calculus

3.1 What happens when we initialise the network?

As described in the previous section, we initialise the network with purely random weights


Terminal — Python main.py — 80×24

```
Year 4 Mathematical Exploration 2024: Trainer
Janav Nagapatla and Aiden Lim
All rights reserved

Importing datasets
> (1/2) Training Dataset
> (2/2) Testing Dataset
> Successfully imported 2 out of 2 datasets

Creating network
> Successfully initialised network with random weights

Evaluating initial performance
> Image 4708 / 10000
> Success 11.00%
█ |██████████████████████-------------------------|
```

The result of this entirely random initialisation is that the neural network is simply making a random guess as to the value of the image. This can be seen as the success rate of the network is about 10% - which is 100% divided by the 10 possible classification groups (with variance accounting for randomness).

This is clearly not suitable as we would expect over 90% accuracy from the network.

Therefore, we will be training the network and choose meaningful values for weights and biases.

3.2 How do we rectify the poor performance of the network?

3.2.1 Discovering how poorly the network has performed?

To discover how poorly a network has performed, we must have an expected output and received output of a neurone in the last layer.

The maximum output for any sigmoid neurone is 1, so the expected value for the correct digit must be a 1. Conversely, the expected value for the wrong digits must be a 0.

Therefore, the error of a particular sigmoid neurone is the expected value subtracted from the received output value.

3.2.2 How do we change the output to make it more accurate?

The factors that convert the 784 input neurones into 10 output neurones are each sigmoid neurone in every layer. At a smaller level, the factors are every single weight and bias of every sigmoid neurone. For our network of 784 ⟶ 30 ⟶ 10 neurones, that is $784 \times 30 + 30 \times 10$ weights and $30 + 10$ biases. That is to say, we must tune 23,860 values to change the output neurone.

This may seem complex at first, but it is valuable to break the problem first into a smaller problem. First, we will want to adjust the last layer of the network to make the output as close to the expected output as possible. In calculations, we can simply represent this as the error, as it shows how the outputs need to be adjusted.

We wish to find out how much the error of the output value of the neurone changes when we change the weights and biases of that particular neurone, while keeping the input value identical.

This can be represented as the derivative of the error w. r. t. the weights of the neurone. We can multiply this derivative by a constant and change the weights by this value to get a smaller error.

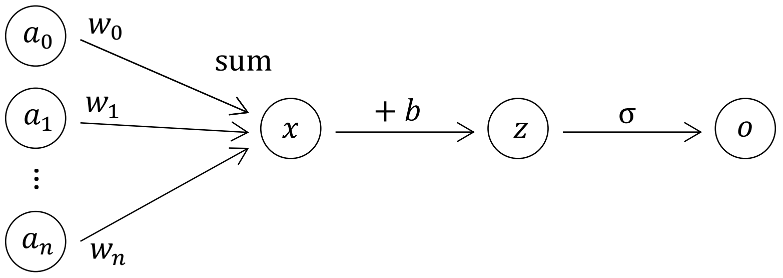
A similar process can be used to update biases, which will be shown later on in the chain rule section too.

After a few cycles of performing differentiation and update of the relevant weights and biases, accuracy of the network eventually reaches ~95%.

3.2.3 Applying Chain Rule to differentiate the sigmoid neurone

As shown previously, by using the derivative of the error w. r. t the weights of the layer, we can update the neural network to produce more accurate output results. We can represent this as $\frac{dE}{dW}$.

The issue with $\frac{dE}{dW}$ is that there is no real way to compute it directly, as there are many transformations that take place between the weight and the error.



Keeping input values the same, first, the weights get summed, have a bias value added to this sum, and go through the sigmoid function. After this, there is still the conversion from output o to the error value. Let us write out the equations for these.

Conventions:

$A = \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{bmatrix}$ (a matrix of all input neurones), $W = \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_n \end{bmatrix}$ (a matrix of all input neurones)

Equations:

$z = A \cdot W + b$ (value of neurone prior to sigmoid activation
$o = \sigma(z)$ (output value of sigmoid neurone to next layer/output evaluation)
$\mathrm{E} = \mathrm{o} - \mathrm{expected}$ (single number value)

Derivatives from equations:

$\frac{dz}{dW} = A$ as b is simply a constant and can be ignored in calculating the derivative
$\frac{do}{dz} = \sigma'(z)$ using prime notation to indicate derivative of a function
$\frac{dE}{do} = 1$ as the expected value is a numerical value and can be ignored in calculating the derivative

As we have three partial derivatives, we can "chain" them together (applying chain rule) and find out our desired derivative: (Sanderson, 2017)

$$\frac{dE}{dW} = \frac{d\mathrm{E}}{d\mathrm{o}} \times \frac{do}{dz} \times \frac{dz}{dW}$$
$$= A \times \sigma'(z)$$

However, we also need the derivative of the error w. r. t the bias, as we will need to update the biases as well. This is quite simple, very similar to the derivative of the error w. r. t the weights.

$$\frac{dE}{db} = \frac{dE}{do} \times \frac{do}{dz} \times \frac{dz}{db}$$

All that has changed is the last term. However, the last term evaluates to 1, as no matter how much the bias value changes, the gradient equation of z is the same (bias is a constant and therefore is irrelevant to the derivative)

Therefore, using the equations we have differentiated for already and that $\frac{dz}{db} = 1$:

$$\frac{dE}{db} = \frac{dE}{do} \times \frac{do}{dz} \times \frac{dz}{db}$$
$$= \sigma'(z)$$

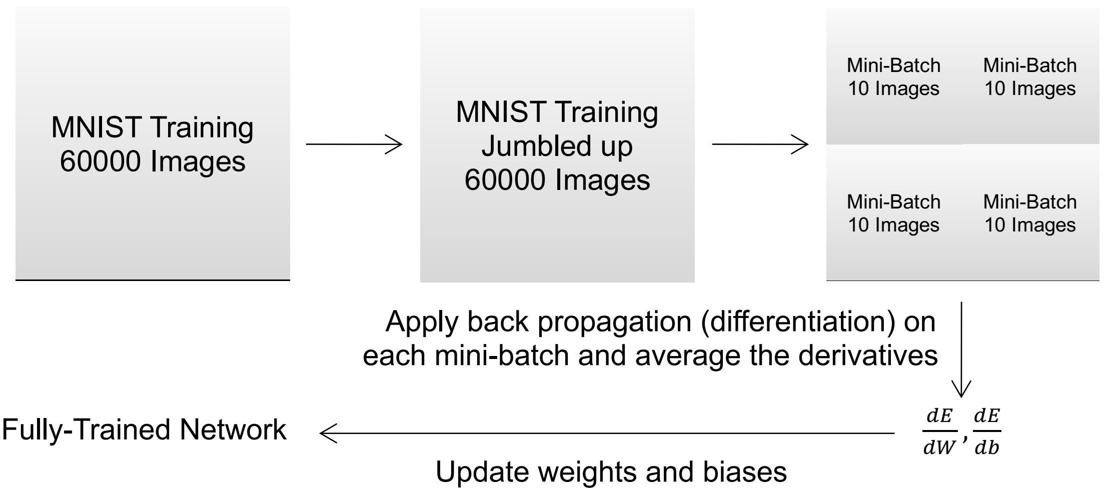
In summary, $\frac{dE}{dW} = A \times \sigma'(\text{z})$ and $\frac{dE}{db} = \sigma'(z)$.

Before moving on, let us differentiate $\sigma(x)$ to find $\sigma'(x)$. As this is not a homework assignment, we have not shown all steps of the derivation in the interest of saving space. However, to differentiate, we simply use chain rule with $(1 + e^{-x})^{-1}$ and the identity $\frac{d}{dx}[e^x] = e^x$ (Lin, 2019)

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$
$$\sigma'(x) = \sigma(x) \times \left(1 - \sigma(x)\right)$$

3.2.3 Applying Stochastic Gradient Descent

As we have the derivatives of the error of any particular neurone with respect to the weights and the biases, we can now perform Stochastic Gradient Descent. (LeCun, 1998)



MNIST Training 60000 Images
MNIST Training Jumbled up 60000 Images
Mini-Batch 10 Images
Mini-Batch 10 Images
Mini-Batch 10 Images
Mini-Batch 10 Images
Apply back propagation (differentiation) on each mini-batch and average the derivatives
$\frac{dE}{dW}, \frac{dE}{db}$
Fully-Trained Network
Update weights and biases

The flowchart illustrates the training process with Stochastic Gradient Descent. Expanded, it can be described as:

1. Scramble the MNIST Dataset of 60,000 to introduce randomness into the training process
2. Divide the MNIST Dataset into 6,000 groups of 10 images
3. Perform differentiation on every image from the last output layer to the first layer
4. Average the derivatives for these 10 images
5. Divide the averaged derivative by a constant to make it smaller in magnitude
6. Subtract the divided derivatives from the current weights and biases of that layer
7. Repeat steps 3 through 6 for all 6,000 groups
8. Repeat steps 1 through 7 a few times to attain a high success rate

4 – Implementing the Algorithm

We do not make the pretense of being extremely proficient at programming, and so while majority of the code was written by ourselves, we have made use of this book by Michael Nielsen: http://neuralnetworksanddeeplearning.com/ to guide us in using linear algebra with Python.

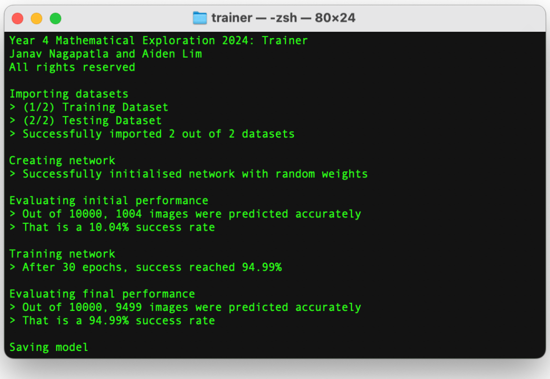
The forward pass is simple to implement and is found in our appendix of code. Simply put: input the image into the first layer of 784 neurones. For every layer thereafter, calculate the new values of every neurone using the sigmoid neurone formula given above and finally, the answer is the index number of the neurone with the highest value out of the ten output neurones.

The backward pass is harder to implement and its source code is also found be found in our appendix. It follows the series of steps outlined in the previous page.

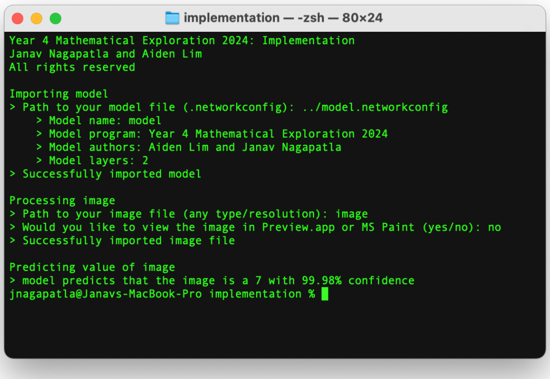
5 – Demonstration of Efficacy

To discover the efficacy of our code, we must run the training program and see the improvement in accuracy: as can be seen, originally, accuracy was 10.04% but reached 94.99% after differentiation and updating via Stochastic Gradient Descent. (test is on a separate dataset from training)

Once we have exported the model in our special file format, we can import it into our implementation and supply our own images. The model predicted an image of a 7 as a 7 with 99.98% confidence.



```
Year 4 Mathematical Exploration 2024: Trainer
Janav Nagapatla and Aiden Lim
All rights reserved

Importing datasets
> (1/2) Training Dataset
> (2/2) Testing Dataset
> Successfully imported 2 out of 2 datasets

Creating network
> Successfully initialised network with random weights

Evaluating initial performance
> Out of 10000, 1004 images were predicted accurately
> That is a 10.04% success rate

Training network
> After 30 epochs, success reached 94.99%

Evaluating final performance
> Out of 10000, 9499 images were predicted accurately
> That is a 94.99% success rate

Saving model
```



```
Year 4 Mathematical Exploration 2024: Implementation
Janav Nagapatla and Aiden Lim
All rights reserved

Importing model
> Path to your model file (.networkconfig): ../model.networkconfig
    > Model name: model
    > Model program: Year 4 Mathematical Exploration 2024
    > Model authors: Aiden Lim and Janav Nagapatla
    > Model layers: 2
> Successfully imported model

Processing image
> Path to your image file (any type/resolution): image
> Would you like to view the image in Preview.app or MS Paint (yes/no): no
> Successfully imported image file

Predicting value of image
> model predicts that the image is a 7 with 99.98% confidence
jnagapatla@Janavs-MacBook-Pro implementation %
```

Upon request (21janav.nagapatla@acsians.acsi.edu.sg), we will be happy to do a demonstration.

6 – Conclusion

The exploration of artificial neural networks through the lens of the MNIST dataset provides valuable insights into how machines can learn from data. This report has covered the fundamental mathematical concepts underpinning neural networks, from the architecture and activation functions to the critical algorithms of backpropagation and gradient descent.

The principles applied to the MNIST dataset serve as a foundation for more complex tasks in image recognition, natural language processing, and beyond.

Interdisciplinary Applications: ANNs have the potential to revolutionize fields such as healthcare, autonomous driving, and financial forecasting, among others.

This report underscores the blend of mathematical rigor and practical experimentation that defines the field of neural networks, opening the door to continued innovation and application across industries.

We thank you for your time in reading our report.

7 - References

Chollet, F. (2017, November 30). *Deep Learning with Python*. Simon and Schuster. http://books.google.ie/books?id=wzozEAAAQBAJ&printsec=frontcover&dq=Deep+Learning+With+Python&hl=&cd=2&source=gbs_api

LeCun, Cortes, & Burges. (1998). MNIST handwritten digit database. Yann LeCun. Retrieved March 22, 2024, from http://yann.lecun.com/exdb/mnist/

LeCun, Y., Bottou, L., Orr, G. B., & Müller, K. R. (1998). Efficient BackProp. *Lecture Notes in Computer Science*, 9–50. https://doi.org/10.1007/3-540-49430-8_2

Lin, H. (2019, December 1). *Deriving the sigmoid derivative via chain and quotient rules*. Data Science. Retrieved March 22, 2024, from https://hausetutorials.netlify.app/posts/2019-12-01-neural-networks-deriving-the-sigmoid-derivative/

Nielsen, M. A. (2015, January 1). *Neural Networks and Deep Learning*. http://books.google.ie/books?id=STDBswEACAAJ&dq=neuralnetworksanddeeplearning&hl=&cd=1&source=gbs_api

Sanderson, G. (2017, November 3). *Backpropagation calculus*. 3Blue1Brown. Retrieved March 22, 2024, from https://www.3blue1brown.com/lessons/backpropagation-calculus

Sanderson, G. (2017, November 3). *What is backpropagation really doing?* 3Blue1Brown. Retrieved March 22, 2024, from https://www.3blue1brown.com/lessons/backpropagation

Sanderson, G. (2017, October 5). *But what is a Neural Network?* 3Blue1Brown. Retrieved March 22, 2024, from https://www.3blue1brown.com/lessons/neural-networks

Anglo-Chinese School (Independent)
Year 4 (2024) Core Mathematics

Mathematical Exploration
Assessment Criteria

Criteria	Descriptor	Score
A. Presentation This criterion assesses the organization and coherence of the exploration. A well-organized exploration contains an introduction, has a rationale (which includes explaining why this topic was chosen), describes the aim of the exploration and has a conclusion. A coherent exploration is logically developed and easy to follow. Graphs, tables and diagrams should accompany the work in the appropriate place and not be attached as appendices to the document.	The exploration does not reach the standard described by the descriptors.	*0*
	The exploration has some coherence.	*1*
	The exploration has some coherence and shows some organization.	*2*
	The exploration is coherent and well organized.	*3*
	The exploration is coherent, well organized, concise.	*4*
B. Mathematical Communication This criterion assess to what extent the student is able to: • use appropriate mathematical language • use multiple forms of mathematical representation, such as formulae, diagrams, tables, charts, graphs and models, where appropriate. • used a deductive method and set out proofs logically where appropriate.	The exploration does not reach the standard described by the descriptors below.	*0*
	The exploration contains some relevant mathematical communication, which is partially appropriate.	*1*
	The exploration contains some relevant appropriate mathematical communication.	*2*
	The mathematical communication is relevant, appropriate and is mostly consistent.	*3*
	The mathematical communication is relevant, appropriate and consistent throughout.	*4*
C. Personal Engagement This criterion assesses the extent to which the student engages with the exploration and makes it his or her own. Personal engagement may be recognized in different attributes and skills. These include thinking independently and/or creatively, addressing personal interest and presenting mathematical ideas in your own way.	The exploration does not reach the standard described by the descriptors below.	*0*
	There is evidence of some personal engagement.	*1*
	There is evidence of significant personal engagement.	*2*
	There is evidence of outstanding personal engagement.	*3*
D. Reflection This criterion assesses how the student reviews, analyses and evaluates the exploration. Although reflection may be seen in the conclusion to the exploration, it may also be found throughout the exploration.	The exploration does not reach the standard described by the descriptors below.	*0*
	There is evidence of limited reflection.	*1*
	There is evidence of meaningful reflection.	*2*
	There is substantial evidence of critical reflection.	*3*
E. Use of Mathematics This criterion assesses to what extent and how well students use mathematics in the exploration. Students are expected to produce work that is commensurate with the level of the course. The mathematics explored should either be part of the syllabus, or at a similar level or beyond. It should <u>not</u> be completely based on mathematics listed in prior learning. A piece of mathematics can be regarded as correct even if there are occasional minor errors as long as they do not detract from the flow of the mathematics or lead to an unreasonable outcome.	The exploration does not reach the standard described by the descriptors below.	*0*
	Some relevant mathematics is used.	*1*
	Some relevant mathematics is used. Limited understanding is demonstrated.	*2*
	Relevant mathematics commensurate with the level of the course is used. Limited understanding is demonstrated.	*3*
	Relevant mathematics commensurate with the level of the course is used. The mathematics explored is partially correct. Some knowledge and understanding are demonstrated.	*4*
	Relevant mathematics commensurate with the level of the course is used. The mathematics explored is mostly correct. Good knowledge and understanding are demonstrated.	*5*
	Relevant mathematics commensurate with the level of the course is used. The mathematics explored is correct. Thorough knowledge and understanding are demonstrated.	*6*