Using the notation for mini-batch gradient descent. To what of the following does $a^{[2]\{4\}(3)}$ correspond? 1 point

- ◯ The activation of the third layer when the input is the fourth example of the second mini-batch.
- ◯ The activation of the second layer when the input is the third example of the fourth mini-batch.
- ◯ The activation of the fourth layer when the input is the second example of the third mini-batch.
- ◯ The activation of the second layer when the input is the fourth example of the third mini-batch.

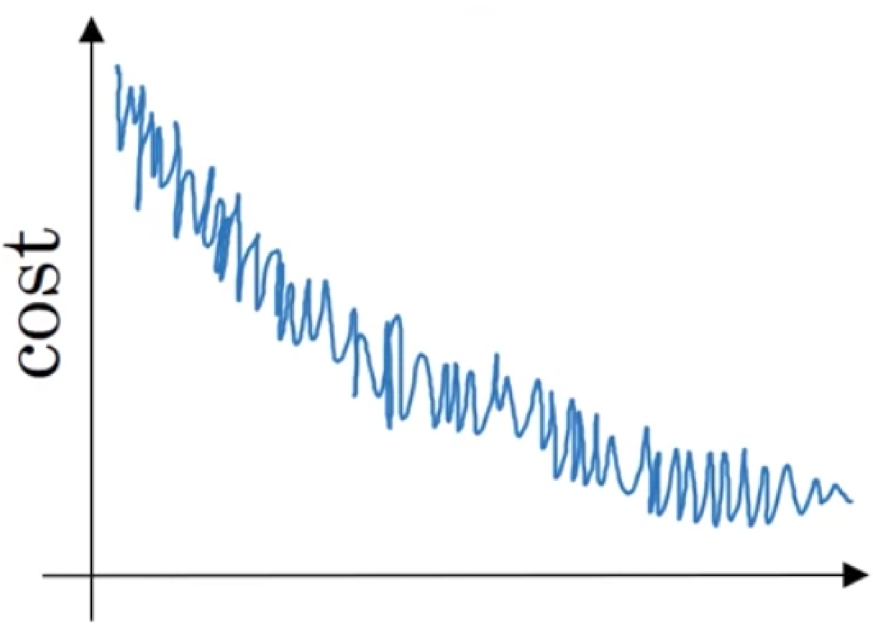
2.Which of these statements about mini-batch gradient descent do you agree with? 1 point

- ◯ Training one epoch (one pass through the training set) using mini-batch gradient descent is faster than training one epoch using batch gradient descent.
- ◯ You should implement mini-batch gradient descent without an explicit for-loop over different mini-batches, so that the algorithm processes all mini-batches at the same time (vectorization).
- ◯ One iteration of mini-batch gradient descent (computing on a single mini-batch) is faster than one iteration of batch gradient descent.

3.Why is the best mini-batch size usually not 1 and not m, but instead something in-between? Check all that are true. 1 point

- ☐ If the mini-batch size is 1, you lose the benefits of vectorization across examples in the mini-batch.
- ☐ If the mini-batch size is m, you end up with stochastic gradient descent, which is usually slower than mini-batch gradient descent.
- ☐ If the mini-batch size is m, you end up with batch gradient descent, which has to process the whole training set before making progress.
- ☐ If the mini-batch size is 1, you end up having to process the entire training set before making any progress.

4.While using mini-batch gradient descent with a batch size larger than 1 but less than m, the plot of the cost function $J$ looks like this: 1 point




You notice that the value of $J$ is not always decreasing. Which of the following is the most likely reason for that?

- ◯ The algorithm is on a local minimum thus the noisy behavior.
- ◯ A bad implementation of the backpropagation process, we should use gradient check to debug our implementation.

- ◯ In mini-batch gradient descent we calculate $J(\hat{y}^{\{t\}}, y^{\{t\}})$ thus with each batch we compute over a new set of data.
- ◯ You are not implementing the moving averages correctly. Using moving averages will smooth the graph.

5.Suppose the temperature in Casablanca over the first two days of January are the same: 1 point

Jan 1st: $\theta_1 = 10^o C$

Jan 2nd: $\theta_2 = 10^o C$

(We used Fahrenheit in the lecture, so we will use Celsius here in honor of the metric world.)

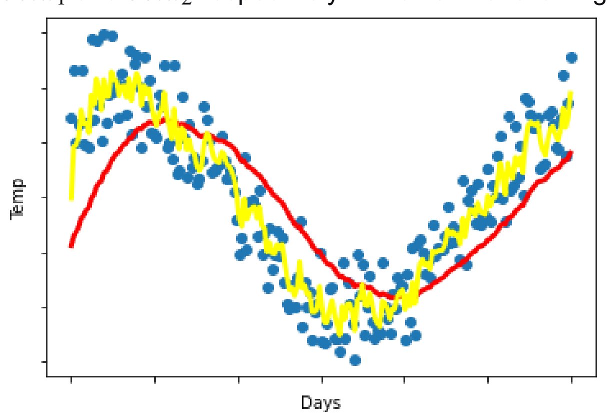
Say you use an exponentially weighted average with $\beta = 0.5$ to track the temperature: $v_0 = 0$, $v_t = \beta v_{t-1} + (1-\beta)\theta_t$. If $v_2$ is the value computed after day 2 without bias correction, and $v_2^{corrected}$ is the value you compute with bias correction. What are these values? (You might be able to do this without a calculator, but you don't actually need one. Remember what bias correction is doing.)

- ◯ $v_2 = 7.5$, $v_2^{corrected} = 7.5$
- ◯ $v_2 = 10$, $v_2^{corrected} = 10$
- ◯ $v_2 = 10$, $v_2^{corrected} = 7.5$
- ◯ $v_2 = 7.5$, $v_2^{corrected} = 10$

6.Which of these is NOT a good learning rate decay scheme? Here, t is the epoch number. 1 point

- ◯ $\alpha = e^t \alpha_0$
- ◯ $\alpha = \frac{1}{\sqrt{t}} \alpha_0$
- ◯ $\alpha = 0.95^t \alpha_0$
- ◯ $\alpha = \frac{1}{1+2*t} \alpha_0$

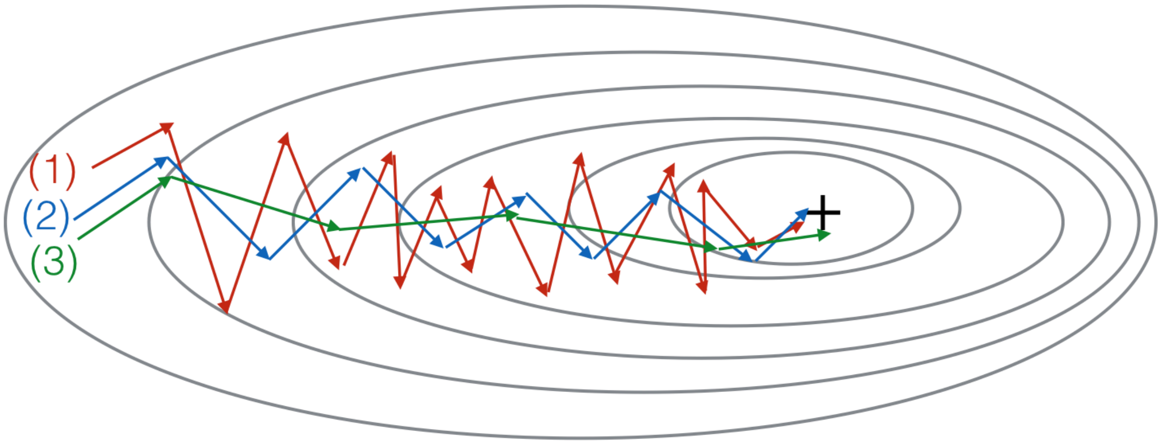
7.You use an exponentially weighted average on the London temperature dataset. You use the following to track the temperature: $v_t = \beta v_{t-1} + (1-\beta)\theta_t$. The yellow and red lines were computed using values $beta_1$ and $beta_2$ respectively. Which of the following are true? 1 point




- ◯ $\beta_1 = \beta_2$.
- ◯ $\beta_1 < \beta_2$.
- ◯ $\beta_1 = 0, \beta_2 > 0$.

○ $\beta_1 > \beta_2$.

8. Consider this figure: 1 point




These plots were generated with gradient descent; with gradient descent with momentum ($\beta$ = 0.5); and gradient descent with momentum ($\beta$ = 0.9). Which curve corresponds to which algorithm?

○ (1) is gradient descent with momentum (small $\beta$), (2) is gradient descent with momentum (small $\beta$), (3) is gradient descent

○ (1) is gradient descent. (2) is gradient descent with momentum (large $\beta$) . (3) is gradient descent with momentum (small $\beta$)

○ (1) is gradient descent with momentum (small $\beta$). (2) is gradient descent. (3) is gradient descent with momentum (large $\beta$)

○ (1) is gradient descent. (2) is gradient descent with momentum (small $\beta$). (3) is gradient descent with momentum (large $\beta$)

9.Suppose batch gradient descent in a deep network is taking excessively long to find a value of the parameters that achieves a small value for the cost function $\mathrm{J}(W^{[1]}, b^{[1]}, ..., W^{[L]}, b^{[L]})$. Which of the following techniques could help find parameter values that attain a small value for $\mathrm{J}$ ? (Check all that apply) 1 point

☐ Try mini-batch gradient descent.

☐ Normalize the input data.

☐ Try using Adam.

☐ Try initializing the weight at zero.

10.Which of the following statements about Adam is *False*? 1 point

○ Adam combines the advantages of RMSProp and momentum

○ Adam should be used with batch gradient computations, not with mini-batches.

○ We usually use "default" values for the hyperparameters $\beta_1, \beta_2$ and $\varepsilon$ in Adam ($\beta_1 = 0.9$, $\beta_2 = 0.999$, $\varepsilon = 10^{-8}$)

○ The learning rate hyperparameter $\alpha$ in Adam usually needs to be tuned.