UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN
Department of Electrical and Computer Engineering

ECE 417 Multimedia Signal Processing
Fall 2023

EXAM 3

Friday, December 8, 2023, 1:30-4:30pm

- This is a CLOSED BOOK exam.
- You are permitted three sheets of handwritten notes, 8.5x11.
- Calculators and computers are not permitted.
- Don't simplify explicit numerical expressions.
- If you're taking the exam online, you will need to have your webcam turned on. Your exam will appear on Gradescope at exactly 1:30pm; you will need to photograph and upload your answers by exactly 4:30pm.
- There are a total of 200 points in the exam. Each problem specifies its point total. Plan your work accordingly.
- You must SHOW YOUR WORK to get full credit.
- This exam contains roughly 17% material from the first third, and 17% material from the second third, and 66% material from the last third of the course, of course.

Name: _____

NetID: _____

Linear Algebra: If $\mathbf{A}$ is tall and thin, with full column rank, then

$$\mathbf{A}^{\dagger}\mathbf{b} = \text{argmin}_{\mathbf{v}}\|\mathbf{b} - \mathbf{A}\mathbf{v}\|^2 = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{b}$$

If $\mathbf{A}$ is short and fat, with full row rank, then

$$\mathbf{A}^{\dagger} = \mathbf{A}^T(\mathbf{A}\mathbf{A}^T)^{-1}$$

Orthogonal projection of $\mathbf{x}$ onto the columns of $\mathbf{A}$ is $\mathbf{x}_{\perp} = \mathbf{A}\mathbf{A}^{\dagger}\mathbf{x}$. Orthogonal projection onto the rows of $\mathbf{A}$ is $\mathbf{x}_{\perp} = \mathbf{A}^{\dagger}\mathbf{A}\mathbf{x}$.

Image Interpolation

$$y[n_1, n_2] = \begin{cases} x\left[\frac{n_1}{U}, \frac{n_2}{U}\right] & \frac{n_1}{U}, \frac{n_2}{U} \text{ both integers} \\ 0 & \text{otherwise} \end{cases}, \qquad z[n_1, n_2] = h[n_1, n_2] * y[n_1, n_2]$$

$$h_{\text{rect}}[n_1, n_2] = \begin{cases} 1 & 0 \le n_1, n_2 < U \\ 0 & \text{otherwise} \end{cases}, \qquad h_{\text{tri}}[n] = \begin{cases} \left(1 - \frac{|n_1|}{U}\right)\left(1 - \frac{|n_2|}{U}\right) & -U \le n_1, n_2 \le U \\ 0 & \text{otherwise} \end{cases}$$

$$h_{\text{sinc}}[n_1, n_2] = \frac{sin(\pi n_1/U)}{\pi n_1/U}\frac{sin(\pi n_2/U)}{\pi n_2/U}$$

Barycentric Coordinates

$$\begin{bmatrix} x_1 \\ x_2 \\ 1 \end{bmatrix} = \beta_1\mathbf{x}_1 + \beta_2\mathbf{x}_2 + \beta_3\mathbf{x}_3 = \begin{bmatrix} x_{1,1} & x_{1,2} & x_{1,3} \\ x_{2,1} & x_{2,2} & x_{2,3} \\ 1 & 1 & 1 \end{bmatrix}\begin{bmatrix} \beta_1 \\ \beta_2 \\ \beta_3 \end{bmatrix}$$

DTFT, DFT, STFT, Griffin-Lim

$$X(\omega) = \sum_n x[n]e^{-j\omega n}, \qquad x[n] = \frac{1}{2\pi}\int_{-\pi}^{\pi} X(\omega)e^{j\omega n}d\omega$$

$$X[k] = \sum_{n=0}^{N-1} x[n]e^{-j\frac{2\pi kn}{N}}, \qquad x[n] = \frac{1}{N}\sum_{k=0}^{N-1} X[k]e^{j\frac{2\pi kn}{N}}$$

$$X_m(\omega) = \sum_n w[n-m]x[n]e^{-j\omega(n-m)}, \quad x[n] = \frac{\sum_m \frac{1}{N}\sum_{k=0}^{N-1} X_m\left(\frac{2\pi k}{N}\right)e^{j\frac{2\pi k(n-m)}{N}}}{\sum_m w[n-m]}$$

$$X_t(\omega) \leftarrow \text{STFT}\left\{\text{ISTFT}\left\{X_t(\omega)\right\}\right\}, \qquad X_t(\omega) \leftarrow M_t(\omega)e^{j\angle X_t(\omega)}$$

Neural Nets

$$h_{i,k}^{(\ell)} = g(z_{i,k}^{(\ell)}), \qquad z_{i,k}^{(\ell)} = b_k^{(\ell)} + \sum_{j=1}^{p} w_{k,j}^{(\ell)}h_{i,j}^{(\ell-1)}$$

$$\frac{d\mathcal{L}}{dz_{i,k}^{(\ell)}} = \frac{d\mathcal{L}}{dh_{i,k}^{(\ell)}}\dot{g}(z_{i,k}^{(\ell)}), \qquad \dot{\sigma}(x) = \sigma(x)(1 - \sigma(x))$$

$$\frac{d\mathcal{L}}{dh_{i,j}^{(\ell-1)}} = \sum_k \frac{d\mathcal{L}}{dz_{i,k}^{(\ell)}}w_{k,j}^{(\ell)}, \qquad \frac{d\mathcal{L}}{dw_{k,j}^{(\ell)}} = \sum_i \frac{d\mathcal{L}}{dz_{i,k}^{(\ell)}}h_{i,j}^{(\ell-1)}$$

Viola-Jones

$$II[m,n] = \sum_{m'=1}^{m} \sum_{n'=1^n} I[m',n'], \quad 1 \le m \le M, 1 \le n \le N$$

$$\epsilon_t = \sum_i w_t(x_i)|y_i - h_t(x_i)|, \quad w_{t+1}(x_i) = \beta_t w_t(x_i), \quad \beta_t = \frac{\epsilon_t}{1-\epsilon_t}$$

$$h(x) = \begin{cases} 1 & \sum_t \alpha_t h_t(x) > \frac{1}{2}\sum_t \alpha_t \\ 0 & \text{otherwise} \end{cases}, \quad \alpha_t = -\ln \beta_t$$

Hidden Markov Model

$$\alpha_t(j) = \sum_{i=1}^{N} \alpha_{t-1}(i) a_{i,j} b_j(\boldsymbol{x}_t), \quad 1 \le j \le N, \ 2 \le t \le T, \quad \hat{\alpha}_t(j) = \frac{\sum_{i=1}^{N} \hat{\alpha}_{t-1}(i) a_{i,j} b_j(\boldsymbol{x}_t)}{\sum_{j'=1}^{N}\sum_{i=1}^{N} \hat{\alpha}_{t-1}(i) a_{i,j'} b_{j'}(\boldsymbol{x}_t)}$$

$$\beta_t(i) = \sum_{j=1}^{N} a_{i,j} b_j(\boldsymbol{x}_{t+1}) \beta_{t+1}(j), \quad 1 \le i \le N, \ 1 \le t \le T-1$$

$$\gamma_t(i) = \frac{\alpha_t(i)\beta_t(i)}{\sum_{k=1}^{N} \alpha_t(k)\beta_t(k)}, \quad \xi_t(i,j) = \frac{\alpha_t(i) a_{i,j} b_j(\boldsymbol{x}_{t+1}) \beta_{t+1}(j)}{\sum_{k=1}^{N}\sum_{\ell=1}^{N} \alpha_t(k) a_{k,\ell} b_\ell(\boldsymbol{x}_{t+1}) \beta_{t+1}(\ell)}$$

$$a'_{i,j} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{j=1}^{N}\sum_{t=1}^{T-1} \xi_t(i,j)}, \quad b'_j(k) = \frac{\sum_{t:x_t=k} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)}$$

Gaussians

$$p_{\boldsymbol{X}}(\boldsymbol{x}) = \frac{1}{(2\pi)^{D/2}|\boldsymbol{\Sigma}|^{1/2}} e^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})}$$

$$\boldsymbol{\mu}'_i = \frac{\sum_{t=1}^{T} \gamma_t(i)\boldsymbol{x}_t}{\sum_{t=1}^{T} \gamma_t(i)}, \quad \boldsymbol{\Sigma}'_i = \frac{\sum_{t=1}^{T} \gamma_t(i)(\boldsymbol{x}_t - \boldsymbol{\mu}_i)(\boldsymbol{x}_t - \boldsymbol{\mu}_i)^T}{\sum_{t=1}^{T} \gamma_t(i)}$$

PCA

$$\boldsymbol{X} = [\boldsymbol{x}_1 - \boldsymbol{\mu}, \ldots, \boldsymbol{x}_M - \boldsymbol{\mu}], \quad \boldsymbol{\Sigma} = \frac{1}{M-1}\boldsymbol{X}\boldsymbol{X}^T$$

$$\boldsymbol{\Sigma} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^T, \ \ \boldsymbol{U}^T\boldsymbol{\Sigma}\boldsymbol{U} = \boldsymbol{\Lambda}, \ \ \boldsymbol{U}^T\boldsymbol{U} = \boldsymbol{U}\boldsymbol{U}^T = \boldsymbol{I}, \ \ \boldsymbol{X} = \boldsymbol{U}\boldsymbol{\Lambda}^{1/2}\boldsymbol{V}^T$$

RNN and LSTM

$$\frac{d\mathcal{L}}{dh[n]} = \frac{\partial \mathcal{L}}{\partial h[n]} + \sum_m \frac{d\mathcal{L}}{dh[n+m]} \frac{\partial h[n+m]}{\partial h[n]}$$

$$i[t] = \sigma_g(w_i x[t] + u_i h[t-1] + b_i), \quad o[t] = \sigma_g(w_o x[t] + u_o h[t-1] + b_o), \quad f[t] = \sigma_g(w_f x[t] + u_f h[t-1] + b_f)$$

$$c[t] = f[t]c[t-1] + i[t]\sigma_h\left(w_c x[t] + u_c h[t-1] + b_c\right), \quad h[t] = o[t]\sigma_h(c[t])$$

1. (23 points) Consider an STFT computed with a 200-sample Hamming window $w[n]$, and with 199-sample overlap between neighboring frames. In the time domain, the signal is $x[n] = \delta[n - 490]$. Find the corresponding STFT, $X_m(\omega)$, for all m and ω. You may express your answer in terms of $w[n]$.

Solution:

$$\begin{aligned} X_m(\omega) &= \sum_n w[n-m]x[n]e^{-j\omega(n-m)} \\ &= \sum_{n'} w[n']x[n'+m]e^{-j\omega n'} \\ &= \sum_{n'} w[n']\delta[n'+m-490]e^{-j\omega n'} \\ &= w[490-m]e^{-j\omega(490-m)} \end{aligned}$$

2. (23 points) Consider a discrete-observation HMM with transition probabilities $a_{i,j} = \Pr\{q_t = j | q_{t-1} = i\}$, observation probabilities $b_j(k) = \Pr\{x_t = k | q_t = j\}$, and initial state probabilities $\pi_i = \Pr\{q_1 = i\}$. Suppose that you have already computed $\alpha_t(i) = \Pr\{x_1, \ldots, x_t, q_t = i\}$ for all frames in the range $1 \leq t \leq 8$, but after frame 8, the input becomes unavailable. Since intelligence is the ability to predict the future, you decide that you want to guess what the observations will be two frames later. What is $\Pr\{x_{10} = \ell | x_1, \ldots, x_8\}$? You may write your answer in terms of π_i, $a_{i,j}$, $b_j(k)$, and/or $\alpha_t(i)$ for any values of i, j, k, t that you find to be useful.

Solution:

$$\begin{aligned}\Pr\{x_{10} = \ell | x_1, \ldots, x_8\} &= \frac{\Pr\{x_{10} = \ell, x_1, \ldots, x_8\}}{\Pr\{x_1, \ldots, x_8\}} \\ &= \frac{\sum_i \sum_j \sum_k \alpha_8(i) a_{i,j} a_{j,k} b_k(\ell)}{\sum_i \alpha_8(i)}\end{aligned}$$

3. (22 points) Suppose you have just finished training an adaboost classifier. The adaboost training algorithm has chosen a sequence of feature definitions $x_1, \ldots, x_T$, and a sequence of signs $p_t \in \{-1, 1\}$, thresholds $\theta_t \in \Re$, and weights $\alpha_t \in \Re$ such that the weak classifiers and strong classifier are given by

$$h_t(\boldsymbol{x}) = \begin{cases} 1 & p_t x_t < p_t \theta_t \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

$$h(\boldsymbol{x}) = \begin{cases} 1 & \sum_{t=1}^{T} \alpha_t h_t(\boldsymbol{x}) > \frac{1}{2} \sum_{t=1}^{T} \alpha_t \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

Now, suppose that you wish to implement your adaboost classifier using a standard two-layer fully-connected network architecture, i.e., you wish to choose some nonlinearities $g_1(\cdot)$ and $g_2(\cdot)$, some weight matrices $\boldsymbol{W}_1$ and $\boldsymbol{W}_2$, and some bias vector $\boldsymbol{b}_1$ and bias scalar b_2 such that

$$h(\boldsymbol{x}) = g_2\left(b_2 + \boldsymbol{W}_2 g_1\left(\boldsymbol{b}_1 + \boldsymbol{W}_1 \boldsymbol{x}\right)\right) \tag{3}$$

Specify $g_1(\cdot)$, $g_2(\cdot)$, $\boldsymbol{W}_1$, $\boldsymbol{W}_2$, $\boldsymbol{b}_1$ and b_2 in terms of p_t, θ_t, and α_t so that Eq. (3) computes the same function as Eq. (2).

Solution: To satisfy

$$h_t(\boldsymbol{x}) = \begin{cases} 1 & p_t x_t < p_t \theta_t \\ 0 & \text{otherwise} \end{cases},$$

we use

$$\begin{aligned} g_1(x) &= u(-x) \\ \boldsymbol{W}_1 &= \text{diag}(p_1, \ldots, p_T) \\ \boldsymbol{b}_1 &= [-p_1\theta_1, \ldots, -p_T\theta_T]^T \end{aligned}$$

or any equivalent. To satisfy

$$h(\boldsymbol{x}) = \begin{cases} 1 & \sum_{t=1}^{T} \alpha_t h_t(\boldsymbol{x}) > \frac{1}{2} \sum_{t=1}^{T} \alpha_t \\ 0 & \text{otherwise} \end{cases},$$

we use

$$\begin{aligned} g_2(x) &= u(x) \\ \boldsymbol{W}_2 &= [\alpha_1, \ldots, \alpha_T] \\ b_2 &= -\frac{1}{2} \sum_{t=1}^{T} \alpha_t \end{aligned}$$

or any equivalent.

4. (22 points) You have a database of N vectors, $\boldsymbol{x}_i \in \Re^D$, $1 \leq i \leq N$. You want to find their pair-wise Mahalanobis distances,

$$d_{\boldsymbol{\Sigma}}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sqrt{(\boldsymbol{x}_i - \boldsymbol{x}_j)^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{x}_i - \boldsymbol{x}_j)}$$

Your friend recommends that you first compute the principal component vectors $\boldsymbol{y}_i = [y_{i,1}, \ldots, y_{i,D}]^T = \boldsymbol{V}^T(\boldsymbol{x}_i - \boldsymbol{\mu})$, where $\boldsymbol{V} = [\boldsymbol{v}_1, \ldots, \boldsymbol{v}_D]$ are the eigenvectors of $\boldsymbol{\Sigma}$, then find the following weighted Euclidean distance:

$$d_{\text{PCA}}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sqrt{\sum_{k=1}^{D} \frac{1}{\lambda_k} (y_{i,k} - y_{j,k})^2},$$

where λ_k is the eigenvalue of $\boldsymbol{\Sigma}$ corresponding to vector $\boldsymbol{v}_k$. Prove that $d_{\text{PCA}}(\boldsymbol{x}_i, \boldsymbol{x}_j) = d_{\boldsymbol{\Sigma}}(\boldsymbol{x}_i, \boldsymbol{x}_j)$.

Solution:

$$\begin{aligned}
d^2_{\text{PCA}}(\boldsymbol{x}_i, \boldsymbol{x}_j) &= \sum_{k=1}^{D} \frac{1}{\lambda_k} (y_{i,k} - y_{j,k})^2 \\
&= (\boldsymbol{y}_i - \boldsymbol{y}_j)^T \boldsymbol{\Lambda}^{-1} (\boldsymbol{y}_i - \boldsymbol{y}_j) \\
&= (\boldsymbol{x}_i - \boldsymbol{x}_j)^T \boldsymbol{V} \boldsymbol{\Lambda}^{-1} \boldsymbol{V}^T (\boldsymbol{x}_i - \boldsymbol{x}_j) \\
&= (\boldsymbol{x}_i - \boldsymbol{x}_j)^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{x}_i - \boldsymbol{x}_j) \\
&= d^2_{\boldsymbol{\Sigma}}(\boldsymbol{x}_i, \boldsymbol{x}_j)
\end{aligned}$$

5. (22 points) A random vector $\boldsymbol{x} = [x_1, x_2]^T \in \Re^2$ is Gaussian with mean $\boldsymbol{\mu} = \mathbf{0}$ and with the following covariance:

$$\boldsymbol{\Sigma} = \frac{1}{5}\begin{bmatrix} 1 & -2 \\ 2 & 1 \end{bmatrix}\begin{bmatrix} 2 & 0 \\ 0 & 0.5 \end{bmatrix}\begin{bmatrix} 1 & 2 \\ -2 & 1 \end{bmatrix}$$

The pdf $p_X(\boldsymbol{x})$ has the following form:

$$p_X(\boldsymbol{x}) = \frac{1}{2\pi|\boldsymbol{\Sigma}|^{1/2}} e^{-\frac{1}{2}\boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{x}}$$

The maximum of the pdf is $p_X(\mathbf{0}) = \frac{1}{2\pi}$. Sketch the set of points $\{\boldsymbol{x} : p_X(\boldsymbol{x}) = \frac{1}{2\pi e^2}\}$. Label the $[x_1, x_2]$ coordinates of the two points on this contour that are farthest from the origin, and of the two points that are closest to the origin.

Solution: $p_X(\boldsymbol{x}) = \frac{1}{2\pi e^2}$ if and only if

$$\frac{1}{2}\boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{x} = 2$$
$$\frac{1}{2}\boldsymbol{y}^T\boldsymbol{\Lambda}^{-1}\boldsymbol{y} = 2$$
$$\frac{y_1^2}{2} + \frac{y_2^2}{0.5} = 4$$

The four extreme points are $(y_1, y_2) = (\pm 4, 0)$ and $(y_1, y_2) = (0, \pm 1)$. Since $y_1 = \boldsymbol{v}_1^T\boldsymbol{x}$ and $y_2 = \boldsymbol{v}_2^T\boldsymbol{x}$, these points correspond to

$$\boldsymbol{x} = \pm 4\boldsymbol{v}_1 = \pm\frac{2\sqrt{2}}{\sqrt{5}}\begin{bmatrix} 1 \\ 2 \end{bmatrix}, \quad \text{(farthest points)}$$
$$\boldsymbol{x} = \pm\boldsymbol{v}_2 = \pm\frac{2\sqrt{2}}{\sqrt{5}}\begin{bmatrix} -2 \\ 1 \end{bmatrix}, \quad \text{(nearest points)}$$

The sketch is an ellipse centered at the origin, with its long axis along $[1, 2]$ and its short axis along $[-2, 1]$.

6. (22 points) An RNN computes an output sequence $\hat{\boldsymbol{Y}} = [\hat{\boldsymbol{y}}_1, \ldots, \hat{\boldsymbol{y}}_T] \in \Re^{K \times T}$ from an input sqeuence $\boldsymbol{X} = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T] \in \Re^{D \times T}$ according to the following rule:

$$\begin{aligned} \boldsymbol{h}_t &= \tanh\left(\boldsymbol{U}\boldsymbol{h}_{t-1} + \boldsymbol{V}\boldsymbol{x}_t\right) \\ \hat{\boldsymbol{y}}_t &= \boldsymbol{W}\boldsymbol{h}_t, \end{aligned}$$

where $\boldsymbol{h}_0 = \boldsymbol{0}$, and where $\boldsymbol{U} \in \Re^{M \times M}$, $\boldsymbol{V} \in \Re^{M \times D}$, and $\boldsymbol{W} \in \Re^{K \times M}$ are some weight matrices. The loss is $\mathcal{L} = \frac{1}{2}\sum_{t=1}^{T} \|\boldsymbol{y}_t - \hat{\boldsymbol{y}}_t\|^2$, where $\boldsymbol{y}_t \in \Re^K$ are some target vectors. Find $\frac{d\mathcal{L}}{d\boldsymbol{h}_t}$ in terms of $\boldsymbol{U}$, $\boldsymbol{V}$, $\boldsymbol{W}$, $\boldsymbol{y}_t$, $\hat{\boldsymbol{y}}_t$, $\boldsymbol{x}_{t+1}$, $\boldsymbol{h}_t$, and/or $\frac{d\mathcal{L}}{d\boldsymbol{h}_{t+1}}$. You may write your answer in terms of $\tanh'(x)$, which is the matrix whose $(i,j)^{\text{th}}$ element is $\frac{d\tanh_i(x)}{dx_j}$.

Solution:

$$\begin{aligned} \frac{d\mathcal{L}}{d\boldsymbol{h}_t} &= \frac{d\mathcal{L}}{d\hat{\boldsymbol{y}}_t}\frac{\partial\hat{\boldsymbol{y}}_t}{\partial\boldsymbol{h}_t} + \frac{d\mathcal{L}}{d\boldsymbol{h}_{t+1}}\frac{\partial\boldsymbol{h}_{t+1}}{\partial\boldsymbol{h}_t} \\ &= (\hat{\boldsymbol{y}}_t - \boldsymbol{y}_t)^T\boldsymbol{W} + \frac{d\mathcal{L}}{d\boldsymbol{h}_{t+1}}\tanh'(\boldsymbol{U}\boldsymbol{h}_t + \boldsymbol{V}\boldsymbol{x}_{t+1})\boldsymbol{U} \end{aligned}$$

7. (22 points) A neural net is a universal approximator only if the hidden layer has a nonlinearity. Without the nonlinearity, any multilayer or recurrent neural net can be rewritten as a simple matrix multiplication. For example, consider the following RNN:

$$\boldsymbol{h}_t = \boldsymbol{U}\boldsymbol{h}_{t-1} + \boldsymbol{V}\boldsymbol{x}_t \tag{4}$$
$$\hat{\boldsymbol{y}}_t = \boldsymbol{W}\boldsymbol{h}_t, \tag{5}$$

where $\boldsymbol{h}_0 = \boldsymbol{0}$, and where $\boldsymbol{U} \in \Re^{M\times M}$, $\boldsymbol{V} \in \Re^{M\times D}$, and $\boldsymbol{W} \in \Re^{K\times M}$ are some weight matrices. This RNN can be rewritten as a linear convolution:

$$\hat{\boldsymbol{y}}_t = \sum_{\tau=0}^{t-1} \boldsymbol{A}_\tau \boldsymbol{x}_{t-\tau} \tag{6}$$

Find the matrices $\boldsymbol{A}_\tau$ in terms of $\boldsymbol{U}$, $\boldsymbol{V}$, $\boldsymbol{W}$, and τ so that Eq. (6) computes the same function as Eq. (5).

Solution:

$$\begin{aligned}\hat{\boldsymbol{y}}_t &= \boldsymbol{W}\boldsymbol{V}\boldsymbol{x}_t + \boldsymbol{W}\boldsymbol{U}\boldsymbol{V}\boldsymbol{x}_{t-1} + \boldsymbol{W}\boldsymbol{U}^2\boldsymbol{V}\boldsymbol{x}_{t-2} + \cdots \\ &= \sum_{\tau=0}^{t-1} \boldsymbol{A}_\tau \boldsymbol{x}_{t-\tau} \\ \boldsymbol{A}_\tau &= \boldsymbol{W}\boldsymbol{U}^\tau\boldsymbol{V}\end{aligned}$$

8. (22 points) Bilinear networks are networks whose outputs depend on the pair-wise products of their inputs. For example, suppose

$$z_i = \boldsymbol{x}_i^T \boldsymbol{W} \boldsymbol{h}_i$$

where $\boldsymbol{W} \in \Re^{D \times M}$, $\boldsymbol{h}_i \in \Re^M$, $\boldsymbol{x}_i \in \Re^D$. Suppose that $\mathcal{L} = -\sum_{i=1}^N \ln z_i$. In terms of $\boldsymbol{x}_i$, $\boldsymbol{h}_i$, and/or z_i, find $\frac{d\mathcal{L}}{dw_{j,k}}$, the derivative of $\mathcal{L}$ with respect to the $(j,k)^{\text{th}}$ element of $\boldsymbol{W}$.

Solution:

$$\begin{aligned}
\frac{d\mathcal{L}}{dw_{j,k}} &= -\sum_{i=1}^N \frac{1}{z_i} \frac{d\mathcal{L}}{dz_i} \frac{\partial z_i}{\partial w_{j,k}} \\
&= -\sum_{i=1}^N \frac{x_{i,j} h_{i,k}}{z_i}
\end{aligned}$$

9. (22 points) Consider an LSTM with zero offsets, and with exclusively linear recurrence, i.e.,

$$i[t] = \sigma(w_i x[t]), \quad o[t] = \sigma(w_o x[t]), \quad f[t] = \sigma(w_f x[t])$$

$$c[t] = f[t]c[t-1] + i[t]\tanh\left(w_c x[t]\right), \quad h[t] = o[t]\tanh(c[t]),$$

where $\sigma(x) = \frac{1}{1+e^{-x}}$ is the logistic sigmoid. Suppose that all the inputs are $x[t] = -\infty$ with one exception: $x[7] = +\infty$. Suppose that $w_o = w_f = -1$, $w_i = w_c = 1$, and $c[0] = 0$. What is $h[100]$?

Solution: When $x[t] = -\infty$,

$$i[t] = \sigma(-\infty) = 0, \quad o[t] = f[t] = \sigma(+\infty) = 1$$

$$c[t] = c[t-1], \quad h[t] = \tanh(c[t])$$

When $x[t] = +\infty$,

$$i[t] = \sigma(\infty) = 1, \quad o[t] = f[t] = \sigma(-\infty) = 0$$

$$c[t] = i[t]\tanh(x[t]) = 1, \quad h[t] = 0$$

So $c[6] = c[5] = \cdots = 0$, but starting at $t = 7$ we have $c[7] = c[8] = \cdots = c[100] = 1$. At $t = 100$ we have

$$h[100] = \tanh(c[100]) = \tanh(1)$$

THIS PAGE IS SCRATCH PAPER.