Time Series and Longitudinal Analyses

CSE545 - Spring 2022

Stony Brook University

H. Andrew Schwartz

Big Data Analytics, The Class

Goal: Generalizations
A *model* or *summarization* of the data.

Data Workflow Frameworks

Hadoop File System ✓
Spark ✓
Streaming ✓
MapReduce ✓
Deep Learning Frameworks ✓

Analytics and Algorithms

Similarity Search ✓
✓ Hypothesis Testing
Regressions->Transformers ✓
Recommendation Systems
Time Series

Big Data Analytics, The Class

Goal: Generalizations
A *model* or *summarization* of the data.

Data Workflow Frameworks

- Hadoop File System ✓
- Spark ✓
- Streaming ✓
- MapReduce ✓
- Deep Learning Frameworks ✓

Analytics and Algorithms

- Similarity Search ✓
- ✓ Hypothesis Testing
- Regressions->Transformers ✓
- Recommendation Systems ✓
- **Time Series**

Intro to Big Data Time-series

Goal: Generalize temporal patterns

Common tasks:

- **Trend Analysis:** Extrapolate patterns over time (typically descriptive).
- **Temporal Relationships:** Correlate Variables over time.
  Does X in year correlate with Y in same year?
  Does X in year 1 correlate with Y in year 2?
- **Forecasting:** Predicting a future event (predictive).
  (contrasts with "cross-sectional" prediction -- predicting a different group)
- **Quasi-Experimental Design:** Evaluate potential causal relationships
  (find relationships more likely than correlation alone, to be causal)

Caution about Causation

X causes Y as opposed to X is associated with Y

Changing X will change the distribution of Y.

X causes Y $\not\leftrightarrow$ Y causes X

Caution about Causation

Spurious Correlations

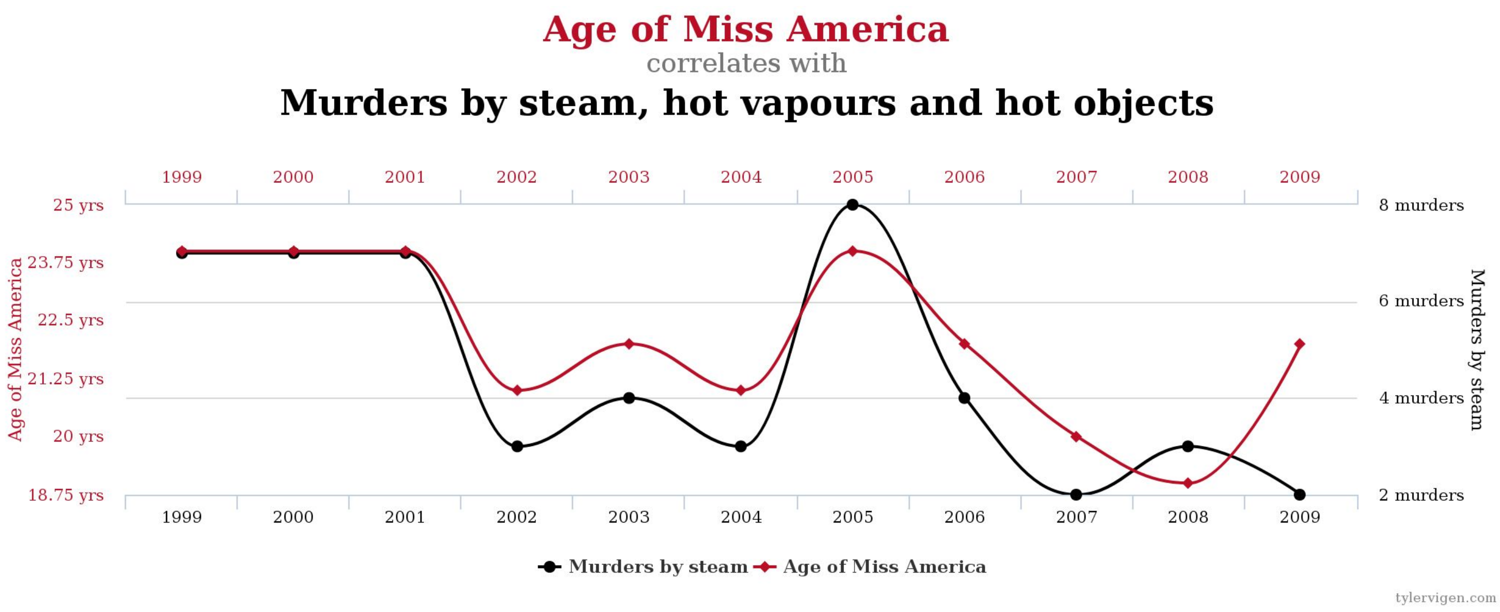
Extremely common in time-series analysis.

http://tylervigen.com/spurious-correlations

Caution about Causation

Spurious Correlations

Extremely common in time-series analysis.



Caution about Causation

X causes Y as opposed to X is associated with Y

Changing X will change the distribution of Y.

X causes Y $\not\leftrightarrow$ Y causes X

$$P(Y=1|X=1) - P(Y=1|X=0)$$

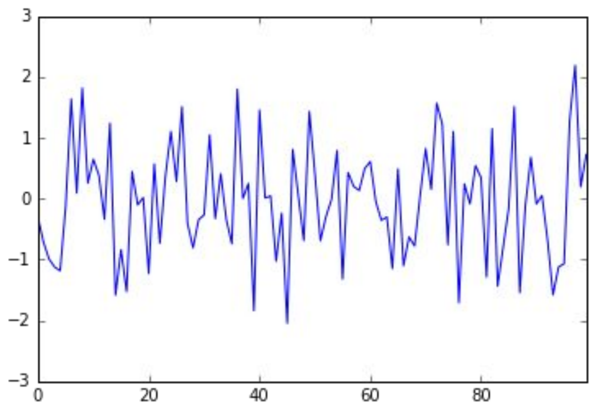
Counterfactual Model: Exposed or Not Exposed: X = 1 or 0

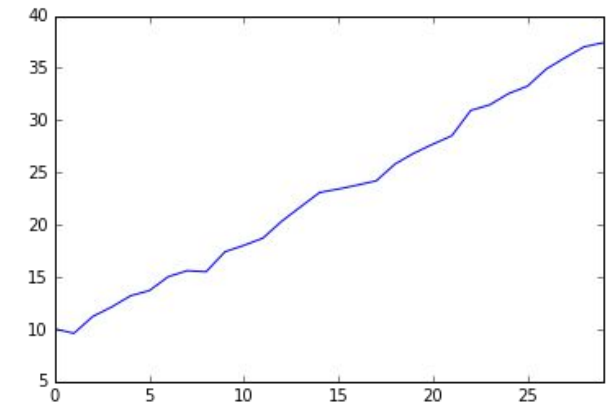
exposure must be random for causality to be concluded

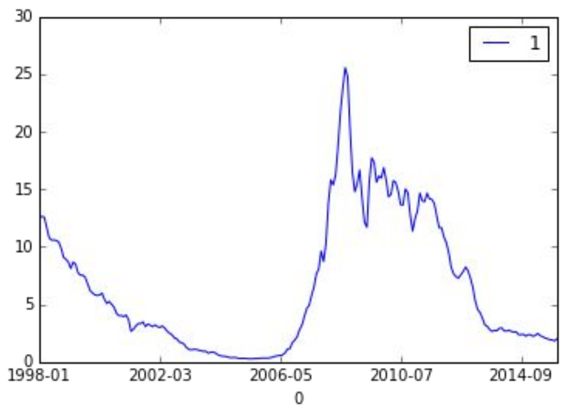
$$Y = \begin{cases} C_0 & \text{if } X = 0 \\ C_1 & \text{if } X = 1 \end{cases}$$

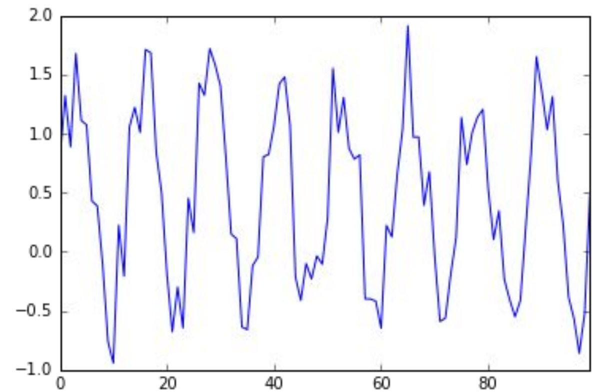
Causal Odds Ratio: $$\frac{\left(\frac{P(C_1=1)}{P(C_1=0)}\right)}{\left(\frac{P(C_0=1)}{P(C_0=0)}\right)}$$

Temporal Patterns

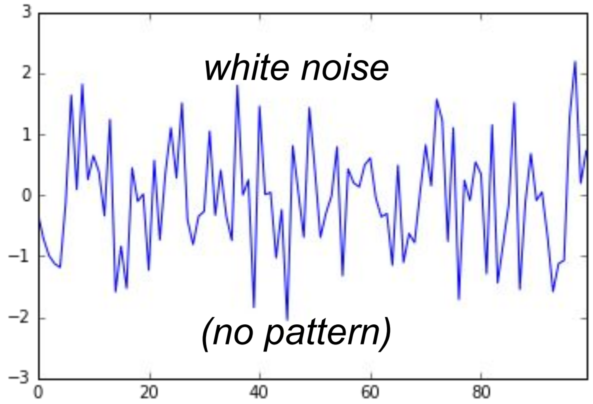


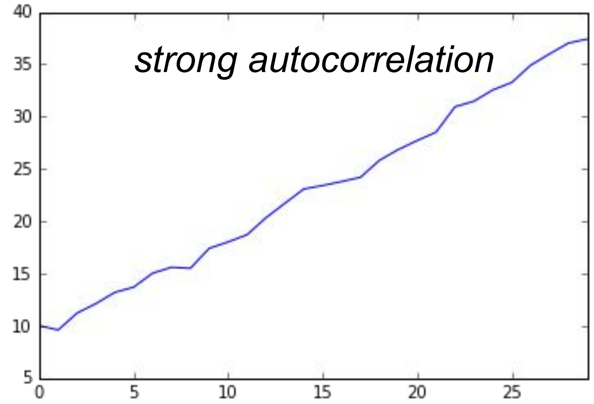


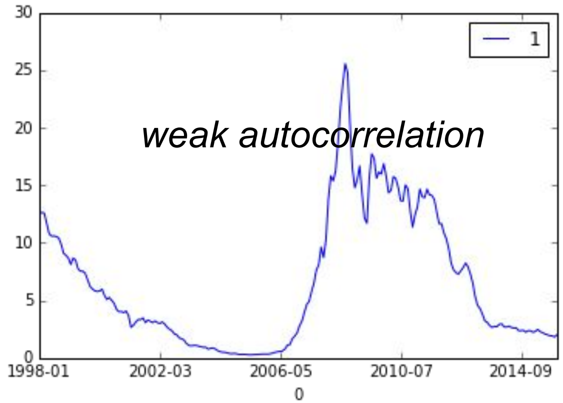


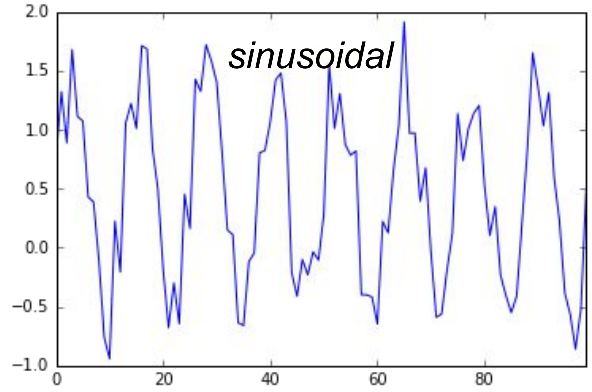


Temporal Patterns









Autoregressive Models (Prediction)

AR Models: $Y_t = f(Y_{t-1}, Y_{t-2}, Y_{t-3}, \ldots, Y_{t-n}, \epsilon_t)$

Linear AR model: $Y_t = \beta_0 + \beta_1 Y_{t-1} + \beta_2 Y_{t-2} + \ldots + \beta_n Y_{t-p} + \epsilon_t$

Autoregressive Models

AR Models: $Y_t = f(Y_{t-1}, Y_{t-2}, Y_{t-3}, \ldots, Y_{t-n}, \epsilon_t)$

Linear AR model: $Y_t = \beta_0 + \beta_1 Y_{t-1} + \beta_2 Y_{t-2} + \ldots + \beta_n Y_{t-p} + \epsilon_t$

Notation:

$$\text{AR}(1)\text{: } \hat{Y}_t = \beta_0 + \beta_1 Y_{t-1}$$
$$\text{AR}(2)\text{: } \hat{Y}_t = \beta_0 + \beta_1 Y_{t-1} + \beta_2 Y_{t-2}$$
$$\text{AR}(3)\text{: } \hat{Y}_t = \beta_0 + \beta_1 Y_{t-1} + \beta_2 Y_{t-2} + \beta_3 Y_{t-3}$$

Autoregressive Models

AR Models: $Y_t = f(Y_{t-1}, Y_{t-2}, Y_{t-3}, \ldots, Y_{t-n}, \epsilon_t)$

Linear AR model: $Y_t = \beta_0 + \beta_1 Y_{t-1} + \beta_2 Y_{t-2} + \ldots + \beta_n Y_{t-p} + \epsilon_t$

Notation:

$$\text{AR}(1)\text{:}\ \hat{Y}_t = \beta_0 + \beta_1 Y_{t-1}$$
$$\text{AR}(2)\text{:}\ \hat{Y}_t = \beta_0 + \beta_1 Y_{t-1} + \beta_2 Y_{t-2}$$
$$\text{AR}(3)\text{:}\ \hat{Y}_t = \beta_0 + \beta_1 Y_{t-1} + \beta_2 Y_{t-2} + \beta_3 Y_{t-3}$$

$$\text{AR}(0)\text{:}\ \hat{Y}_t = \beta_0$$

Moving Average Models

Based on error;   (a “smoothing” technique).

Q: Best estimator of random data (i.e. white noise)?

Moving Average Models

Based on error;   (a “smoothing” technique).

Q: Best estimator of random data (i.e. white noise)?

A: The mean

$$\hat{Y}_t^{MA} = \frac{Y_t + Y_{t-1} + Y_{t-2} + \ldots + Y_{t-p}}{p+1}$$

Moving Average Models

Based on error;   (a "smoothing" technique).

Q: Best estimator of random data (i.e. white noise)?

A: The mean

$$\hat{Y}_t^{MA} = \frac{Y_t + Y_{t-1} + Y_{t-2} + ... + Y_{t-p}}{p+1}$$

Simple Moving Average

Moving Average Models

In a regression model (AR**MA** or ARI**MA**), we consider error terms

$$Y_t = f(\epsilon_t, \epsilon_{t-1}, \epsilon_{t-2}, \epsilon_{t-3}, \ldots)$$

Moving Average Models

In a regression model (AR**MA** or ARI**MA**), we consider error terms

$$Y_t = f(\epsilon_t, \epsilon_{t-1}, \epsilon_{t-2}, \epsilon_{t-3}, \ldots)$$

$$\hat{Y}_t = \mu + \epsilon_t + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + \ldots + \theta_p \epsilon_{t-p}$$

Moving Average Models

In a regression model (AR**MA** or ARI**MA**), we consider error terms

$$Y_t = f(\epsilon_t, \epsilon_{t-1}, \epsilon_{t-2}, \epsilon_{t-3}, \ldots)$$

$$\hat{Y}_t = \mu + \epsilon_t + \theta_1 \boxed{\epsilon_{t-1}} + \theta_2 \boxed{\epsilon_{t-2}} + \ldots + \theta_p \boxed{\epsilon_{t-p}}$$

attributed to “shocks” -- independent, from a normal distribution

Notation:

$$\text{MA}(1)\text{: } \hat{Y}_t = \mu + \epsilon_t + \theta_1 \epsilon_{t-1}$$
$$\text{MA}(2)\text{: } \hat{Y}_t = \mu + \epsilon_t + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2}$$

ARMA Models

AutoRegressive (AR) Moving Average (MA) Model

ARMA(p, q):

$$\hat{Y}_t = \beta_0 + \beta_1 Y_{t-1} + \beta_2 Y_{t-2} + \ldots + \beta_p Y_{t-p} + \\ \epsilon_t + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + \ldots + \theta_q \epsilon_{t-q}$$

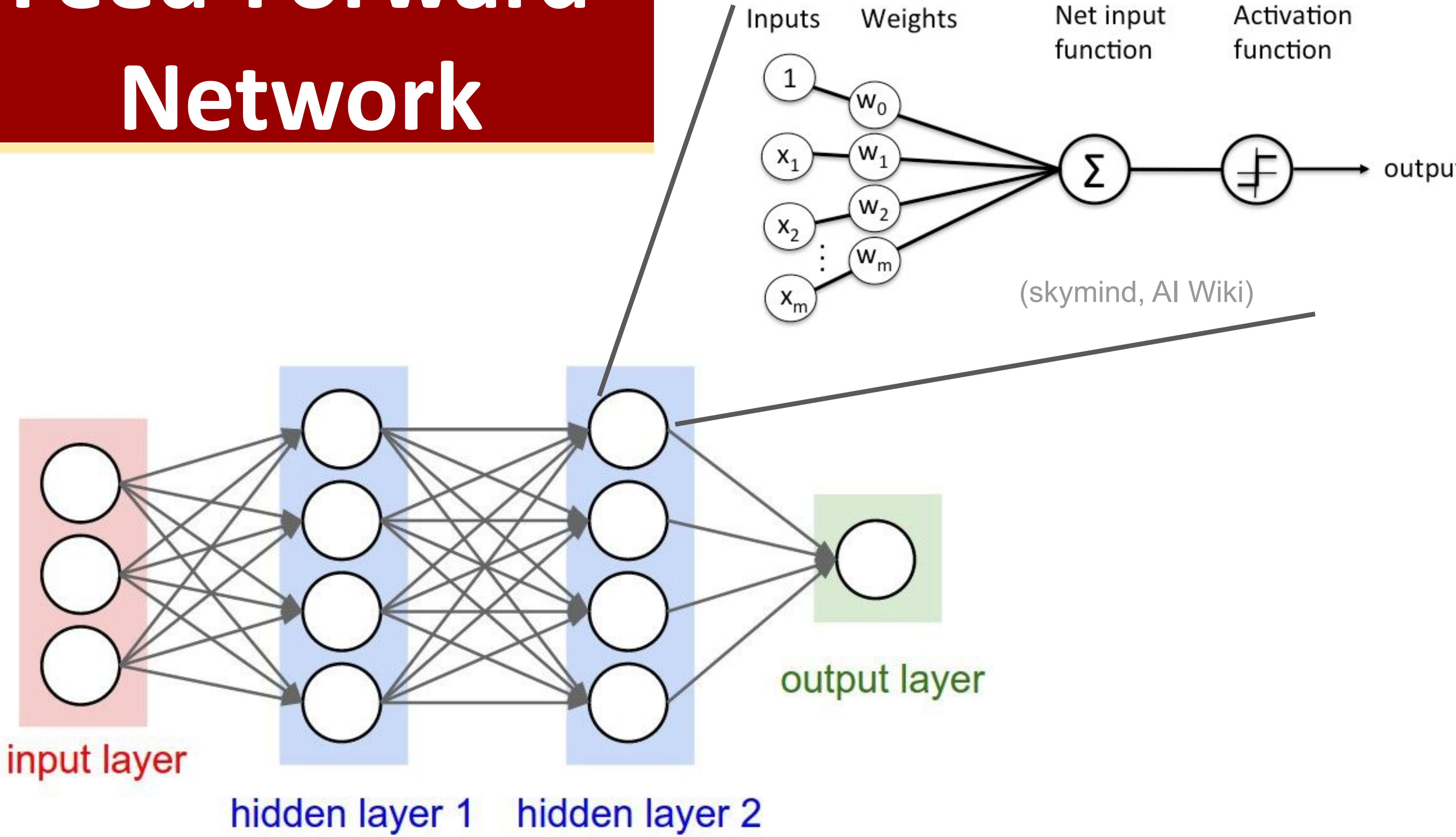
ARMA(1, 1):

$$\hat{Y}_t = \beta_1 Y_{t-1} + \epsilon_t + \theta_1 \epsilon_{t-1}$$

example: Y is sales; error may be effect from coupon or advertising

(credit: Ben Lambert)

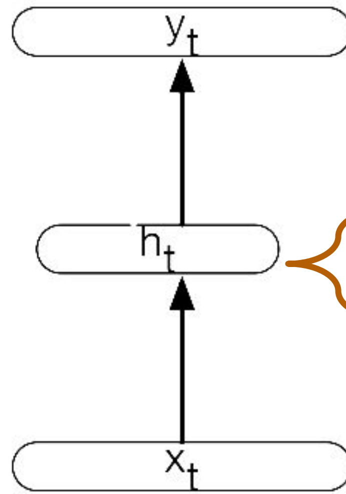
Feed-Forward Network



(skymind, AI Wiki)

~~Recurrent~~ Neural Network

“hidden layer”



$$y_{(t)} = f(h_{(t)}W)$$

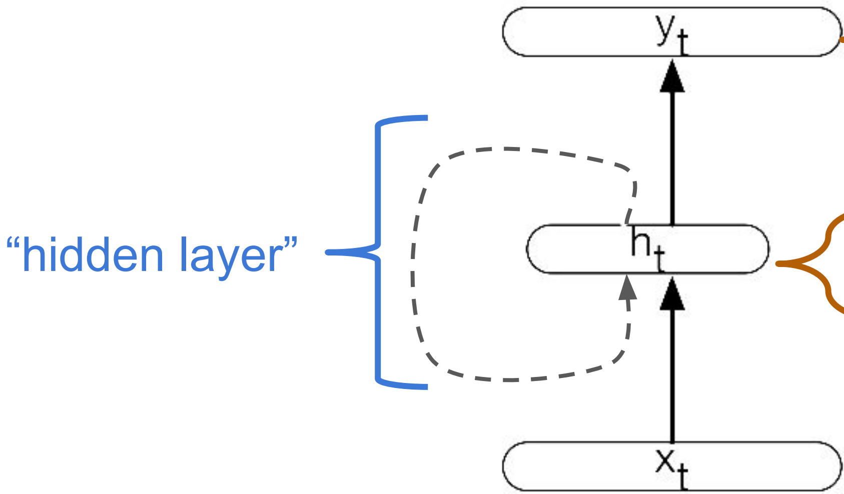
Activation Function

$$h_{(t)} = g(x_{(t)}V)$$

Figure 9.2 Simple recurrent neural network after Elman (Elman, 1990). The hidden layer includes a recurrent connection as part of its input. That is, the activation value of the hidden layer depends on the current input as well as the activation value of the hidden layer from the previous timestep.

(Jurafsky, 2019)

Recurrent Neural Network



$$y_{(t)} = f(h_{(t)}W)$$

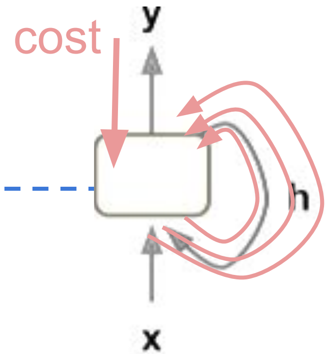
Activation Function

$$h_{(t)} = g(\boldsymbol{h}_{(t-1)}\boldsymbol{U} + x_{(t)}V)$$

Figure 9.2 Simple recurrent neural network after Elman (Elman, 1990). The hidden layer includes a recurrent connection as part of its input. That is, the activation value of the hidden layer depends on the current input as well as the activation value of the hidden layer from the previous timestep.

(Jurafsky, 2019)

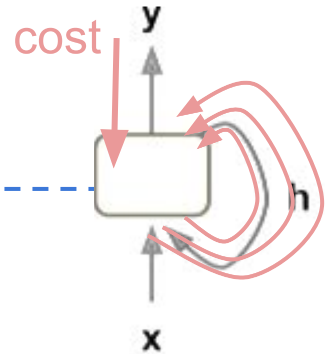
RNN: Optimization



Backward Propagation through Time

```
...
#define forward pass graph:
h(0) = 0
for i in range(1, len(x)):
    h(i) = tf.tanh(tf.matmul(U,h(i-1))+ tf.matmul(W,x(i))) #update hidden state
    y(i) = tf.softmax(tf.matmul(V, h(i))) #update output
...
cost = tf.reduce_mean(-tf.reduce_sum(y*tf.log(y_pred))
```

RNN: Optimization



Backward Propagation through Time

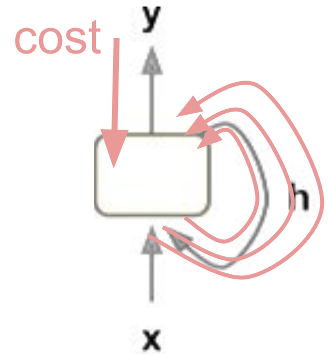
```
...
#define forward pass graph:
h(0) = 0
for i in range(1, len(x)):
    h(i) = tf.tanh(tf.matmul(U,
state
    y(i) = tf.softmax(tf.matmul
...
cost = tf.reduce_mean(-tf.redu
```

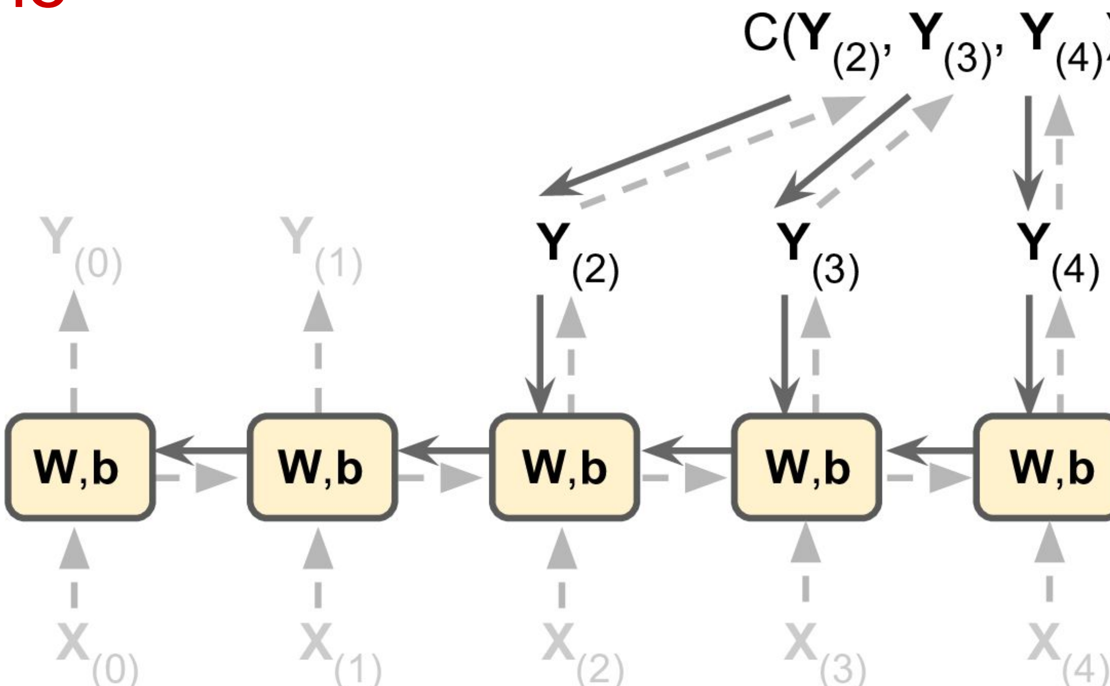
To find the gradient for the overall graph, we use **back propogation,** which *essentially* chains together the gradients for each node (function) in the graph.

With many recursions, the gradients can vanish or explode (become too large or small for floating point operations).

RNN: Optimization

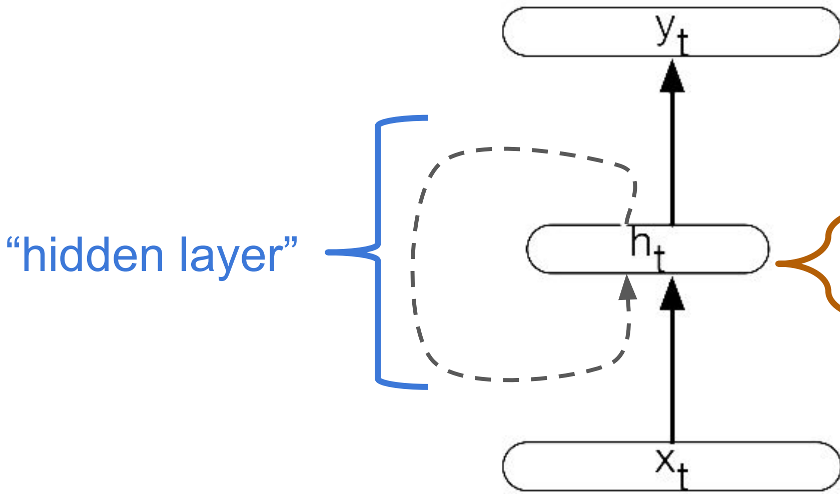
Backward Propagation through Time





(Geron, 2017)

GRU-based RNN



$$y_{(t)} = f(h_{(t)}W)$$

Activation Function

$$h_{(t)} = GRU(\boldsymbol{h}_{(t-1)}, x_{(t)})$$

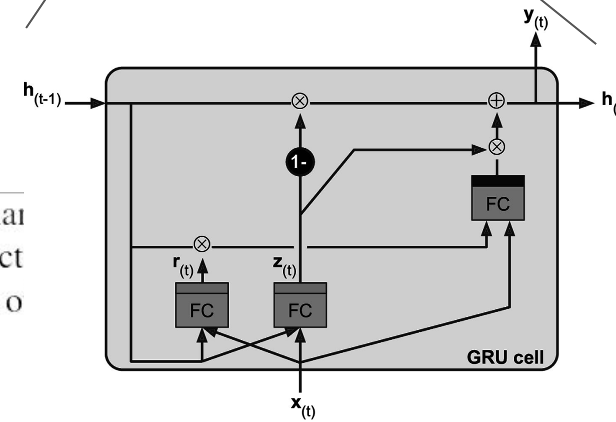


Figure 9.2 Simple recurrent neural network after Elman (Elmai
includes a recurrent connection as part of its input. That is, the act
layer depends on the current input as well as the activation value o
previous timestep.

(Jurafsky, 2019)

Time-Series Applications

- ARMA
  - Economic indicators
  - System performance
  - Trend analysis
    (often situations where there is a general trend and random “shocks”)
- Univariate Models in General
  - Anomaly Detection
  - Forecasting
  - Season Trends
  - Signal Processing

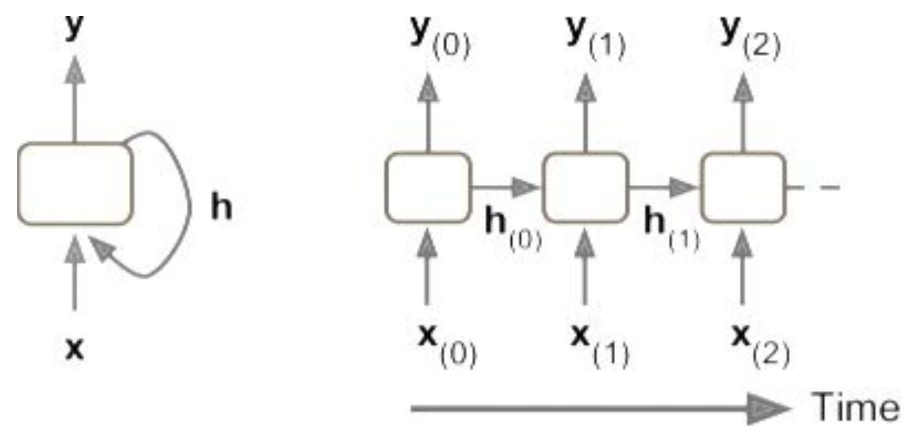
- Integration as predictors within multivariate models

statsmodels.tsa.arima_model

Supplement

How to Addressing Vanishing Gradient?

Dominant approach: Use Long Short Term Memory Networks (LSTM)



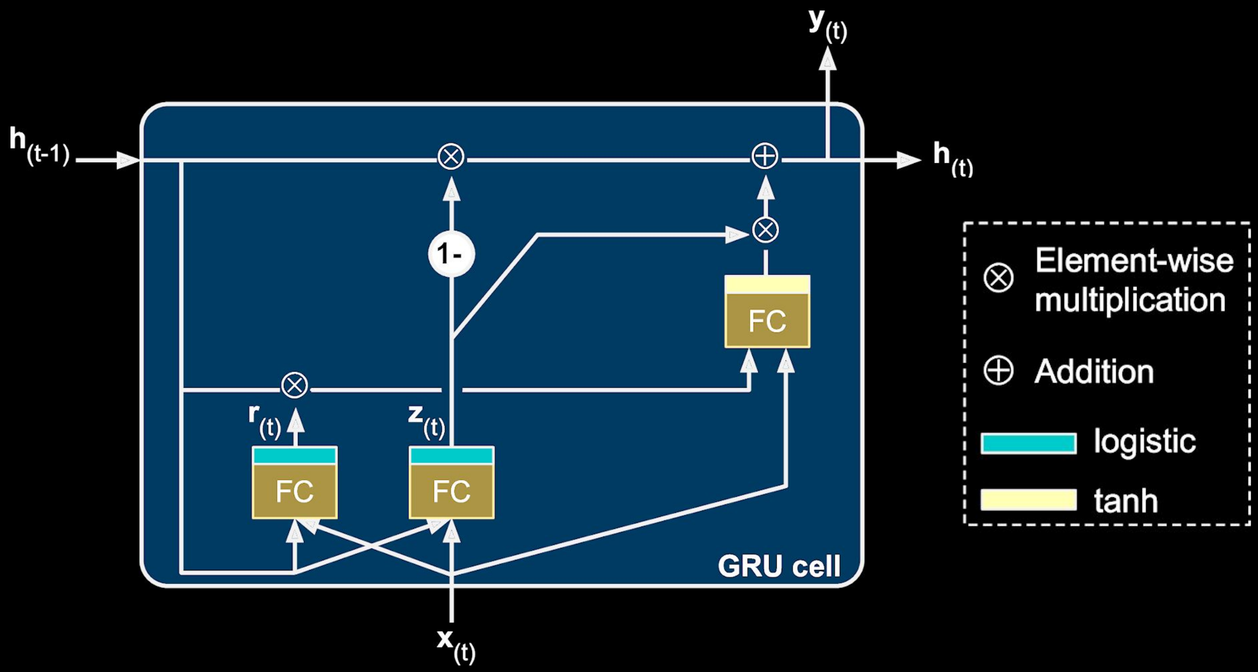
RNN model

"unrolled" depiction

(Geron, 2017)

RNN: The GRU

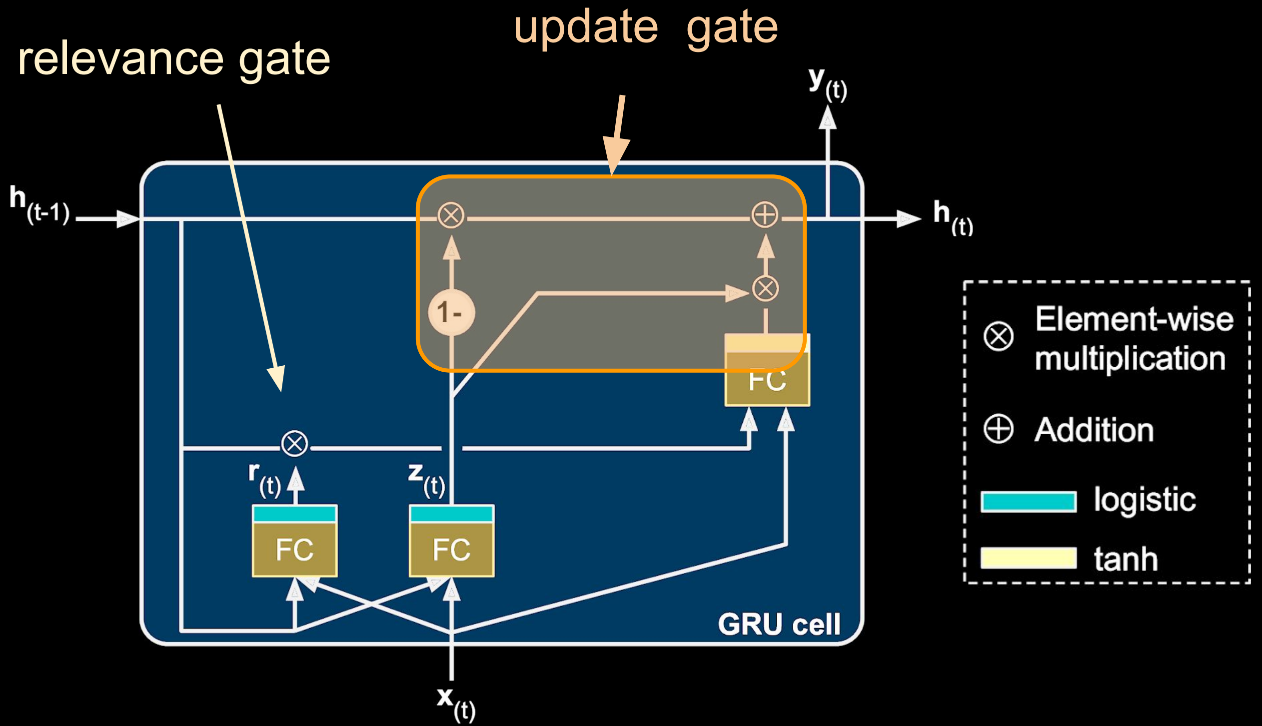
Gated Recurrent Unit



(Geron, 2017)

RNN: The GRU

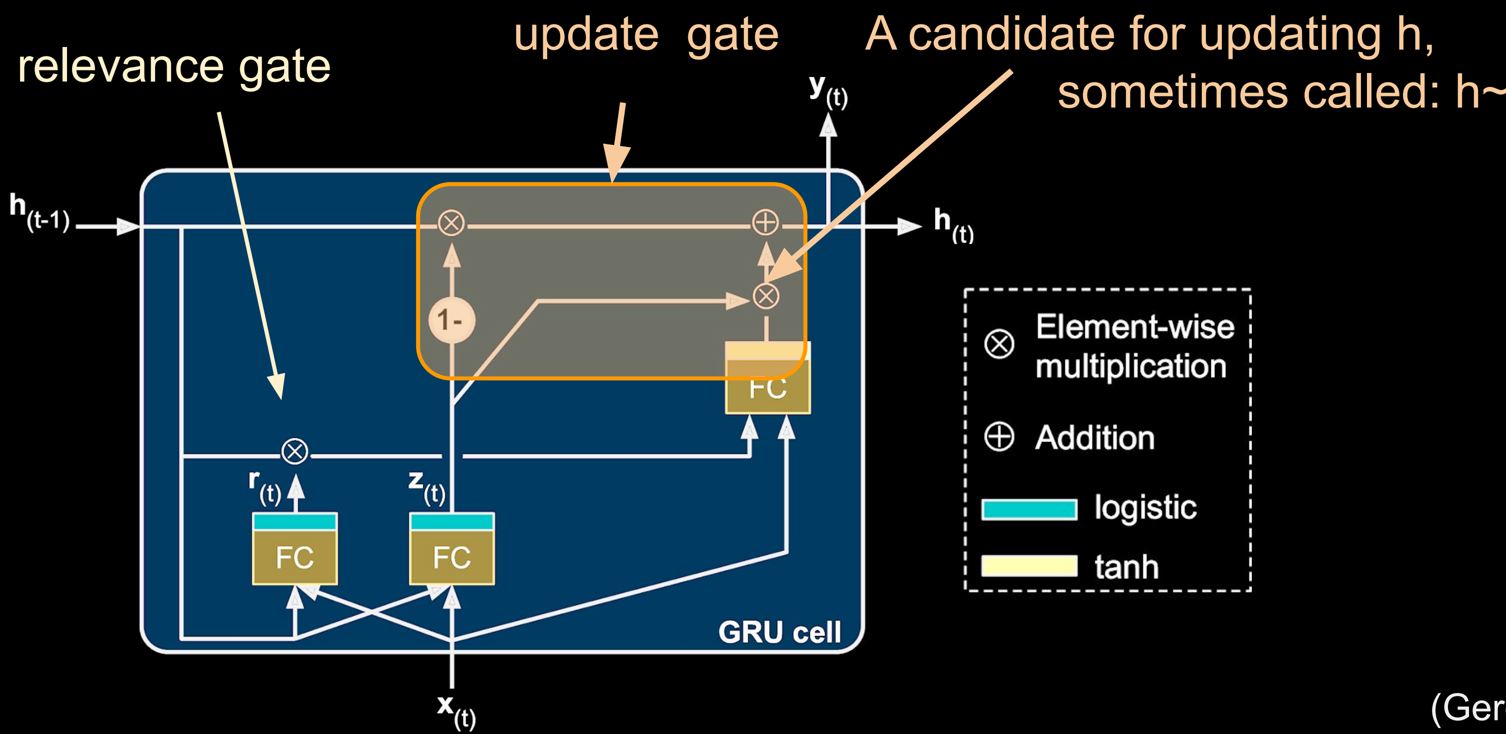
Gated Recurrent Unit



(Geron, 2017)

RNN: The GRU

Gated Recurrent Unit



(Geron, 2017)

RNN: The GRU

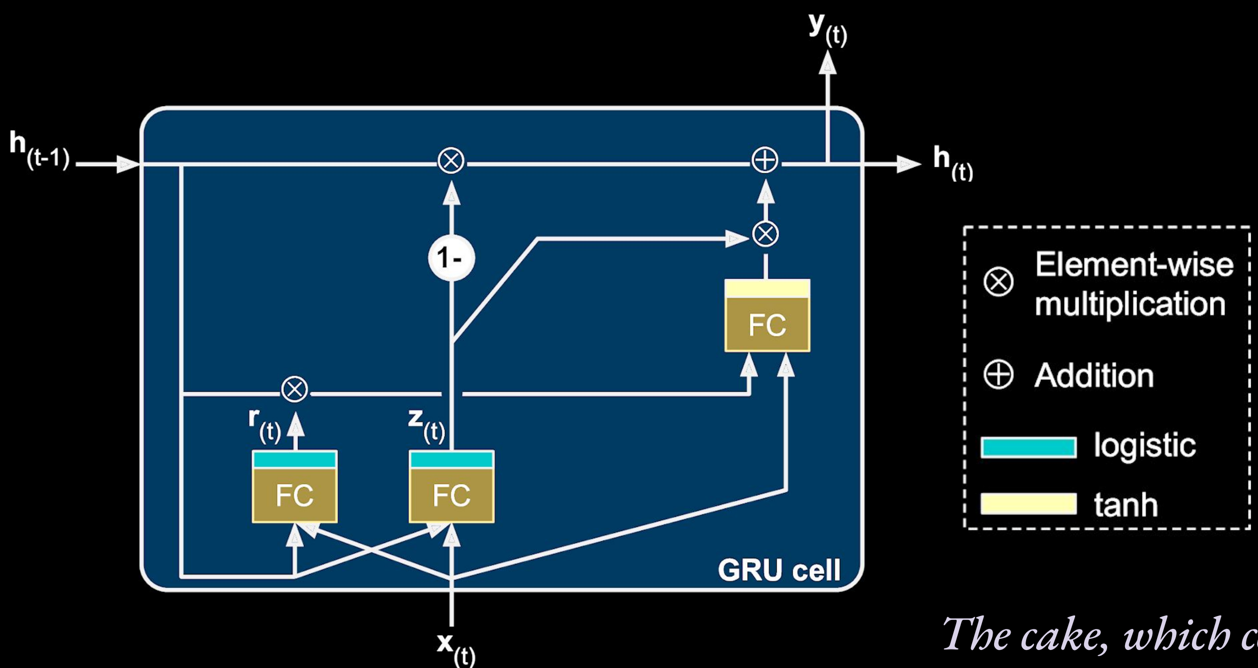
Gated Recurrent Unit

$$\mathbf{z}_{(t)} = \sigma(\mathbf{W}_{xz}{}^{T} \cdot \mathbf{x}_{(t)} + \mathbf{W}_{hz}{}^{T} \cdot \mathbf{h}_{(t-1)} + \mathbf{b}_z)$$

$$\mathbf{r}_{(t)} = \sigma(\mathbf{W}_{xr}{}^{T} \cdot \mathbf{x}_{(t)} + \mathbf{W}_{hr}{}^{T} \cdot \mathbf{h}_{(t-1)} + \mathbf{b}_r)$$

$$\mathbf{g}_{(t)} = \tanh\left(\mathbf{W}_{xg}{}^{T} \cdot \mathbf{x}_{(t)} + \mathbf{W}_{hg}{}^{T} \cdot (\mathbf{r}_{(t)} \otimes \mathbf{h}_{(t-1)}) + \mathbf{b}_g\right)$$

$$\mathbf{h}_{(t)} = \mathbf{z}_{(t)} \otimes \mathbf{h}_{(t-1)} + (1 - \mathbf{z}_{(t)}) \otimes \mathbf{g}_{(t)}$$



The cake, which contained candles, was eaten.

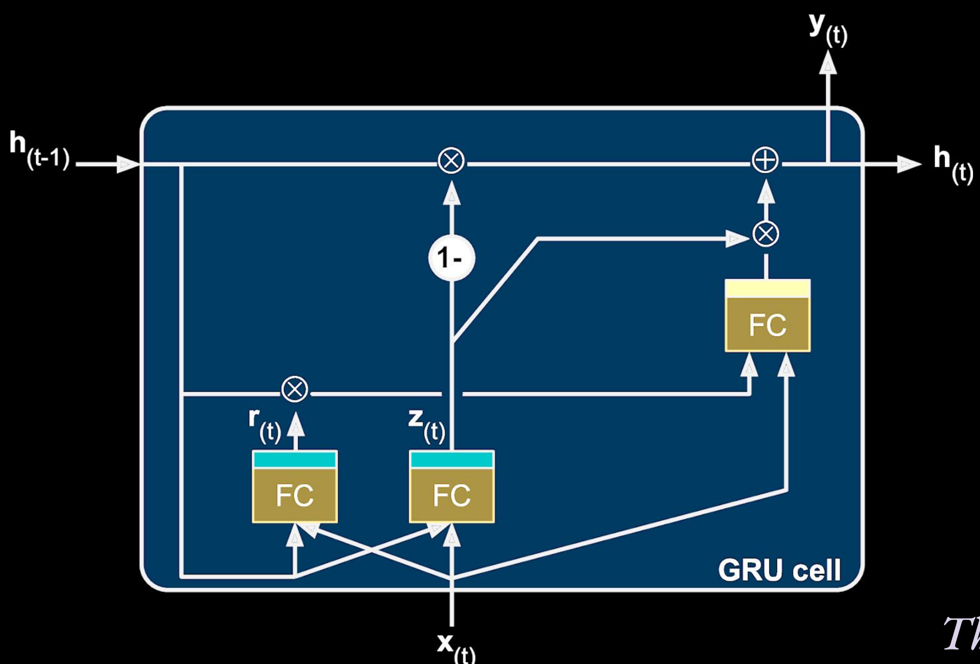
What about the gradient?

$$\mathbf{z}_{(t)} = \sigma(\mathbf{W}_{xz}{}^T \cdot \mathbf{x}_{(t)} + \mathbf{W}_{hz}{}^T \cdot \mathbf{h}_{(t-1)} + \mathbf{b}_z)$$

$$\mathbf{r}_{(t)} = \sigma(\mathbf{W}_{xr}{}^T \cdot \mathbf{x}_{(t)} + \mathbf{W}_{hr}{}^T \cdot \mathbf{h}_{(t-1)} + \mathbf{b}_r)$$

$$\mathbf{g}_{(t)} = \tanh\,(\mathbf{W}_{xg}{}^T \cdot \mathbf{x}_{(t)} + \mathbf{W}_{hg}{}^T \cdot (\mathbf{r}_{(t)} \otimes \mathbf{h}_{(t-1)}) + \mathbf{b}_g)$$

$$\mathbf{h}_{(t)} = \mathbf{z}_{(t)} \otimes \mathbf{h}_{(t-1)} + (1 - \mathbf{z}_{(t)}) \otimes \mathbf{g}_{(t)}$$



The gates (i.e. multiplications based on a logistic) often end up keeping the hidden state exactly (or nearly exactly) as it was. Thus, for most dimensions of h,

$$h_{(t)} \approx h_{(t-1)}$$

The cake, which contained candles, was eaten.

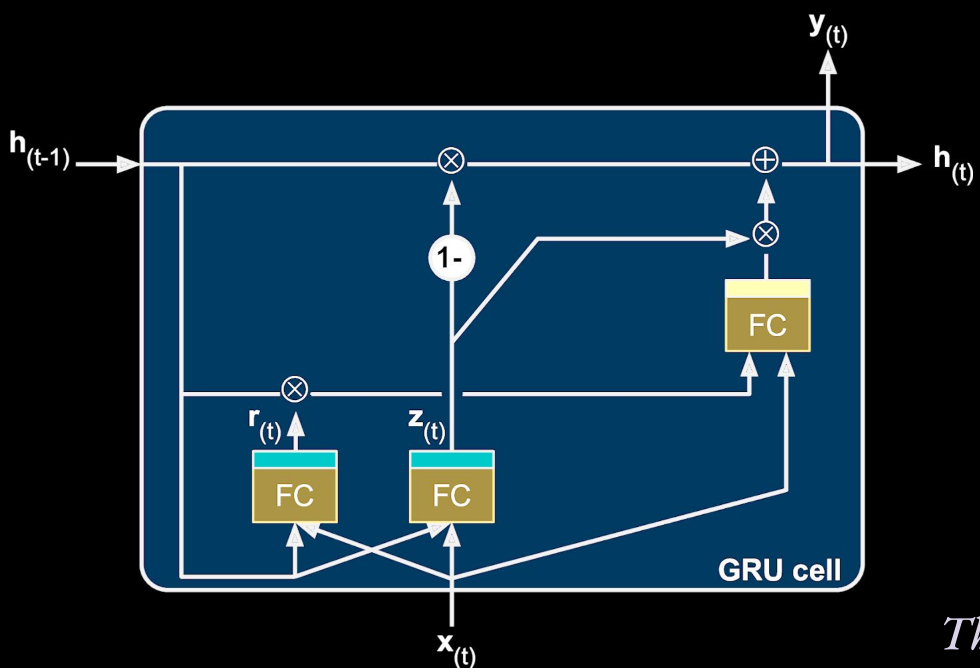
What about the gradient?

$$\mathbf{z}_{(t)} = \sigma(\mathbf{W}_{xz}{}^T \cdot \mathbf{x}_{(t)} + \mathbf{W}_{hz}{}^T \cdot \mathbf{h}_{(t-1)} + \mathbf{b}_z)$$

$$\mathbf{r}_{(t)} = \sigma(\mathbf{W}_{xr}{}^T \cdot \mathbf{x}_{(t)} + \mathbf{W}_{hr}{}^T \cdot \mathbf{h}_{(t-1)} + \mathbf{b}_r)$$

$$\mathbf{g}_{(t)} = \tanh\left(\mathbf{W}_{xg}{}^T \cdot \mathbf{x}_{(t)} + \mathbf{W}_{hg}{}^T \cdot (\mathbf{r}_{(t)} \otimes \mathbf{h}_{(t-1)}) + \mathbf{b}_g\right)$$

$$\mathbf{h}_{(t)} = \mathbf{z}_{(t)} \otimes \mathbf{h}_{(t-1)} + (1 - \mathbf{z}_{(t)}) \otimes \mathbf{g}_{(t)}$$



The gates (i.e. multiplications based on a logistic) often end up keeping the hidden state exactly (or nearly exactly) as it was. Thus, for most dimensions of h,

$$h_{(t)} \approx h_{(t-1)}$$

This tends to keep the gradient from vanishing since the same values will be present through multiple times in backpropagation through time. (The same idea applies to LSTMs but is easier to see here).

The cake, which contained candles, was eaten.

The GRU (LSTM): Zoomed out

Take-Aways

- Simple RNNs are powerful models but they are difficult to train:
  - Just two functions $h_{(t)}$ and $y_{(t)}$ where $h_{(t)}$ is a combination of $h_{(t-1)}$ and $x_{(t)}$.
  - Exploding and vanishing gradients make training difficult to converge.
- LSTM (e.g. GRU cells) solve
  - Hidden states pass from one time-step to the next, allow for long-distance dependencies.
  - Gates are used to keep hidden states from changing rapidly (and thus keeps gradients under control).
  - To train: mini-batch stochastic gradient descent over cross-entropy cost