Sequence Learning

Cost Functions and States

Korbinian Riedhammer

TECHNISCHE HOCHSCHULE NÜRNBERG
GEORG SIMON OHM

Levenshtein

Basic edit distance

- Works for different-length sequences
- Uniform cost for substitution/ insertion/deletion

```
def edit(x, y):
    D = np.zeros((len(x) + 1, len(y) + 1), dtype=int)
    # for the empty word, costs match the length of
    # the other string
    D[0, 1:] = range(1, len(y) + 1)
    D[1:, 0] = range(1, len(x) + 1)

    for i in range(1, len(x) + 1):
        for j in range(1, len(y) + 1):
            delta = 1 if x[i-1] == y[j-1] else 0
            D[i, j] = min(
                D[i-1, j] + 1,
                D[i, j-1] + 1,
                D[i-1, j-1] + delta
            )

    return D[len(x), len(y)]
```

Levenshtein

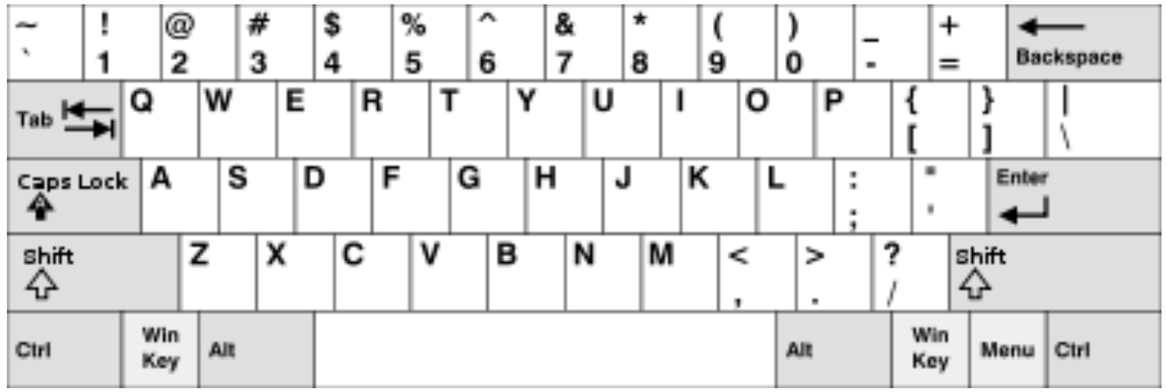
Custom Cost

```
def edit(x, y, cost={'m': 0, 's': 1, 'i': 1, 'd': 1}):
    D = np.zeros((len(x) + 1, len(y) + 1), dtype=int)
    # for the empty word, costs match the length of the
    # other string
    D[0, 1:] = range(1, len(y) + 1)
    D[1:, 0] = range(1, len(x) + 1)
    for i in range(1, len(x) + 1):
        for j in range(1, len(y) + 1):
            delta = cost['m'] if x[i-1] == y[j-1] else cost['s']
            D[i, j] = min(
                D[i-1, j] + cost['d'],
                D[i, j-1] + cost['i'],
                D[i-1, j-1] + delta
            )
    return D[len(x), len(y)]
```

Needleman-Wunsch Algorithm

- Biology: not all edits “cost the same”, gaps typically treated separately
- Gap penalty (eg. -1)
- Similarity (match reward, mismatch penalty)
- DP finds maximum similarity

	A	G	C	T
A	10	−1	−3	−4
G	−1	7	−5	−3
C	−3	−5	9	0
T	−4	−3	0	8



https://commons.wikimedia.org/wiki/File:KB_United_States.svg#/media/File:KB_United_States.svg

Needleman-Wunsch Algorithm

```
def nw(x, y, d, sim):
    D = np.zeros((len(x) + 1, len(y) + 1), dtype=int)

    # for the empty word, costs match the length of the other string
    D[0, 1:] = range(1, len(y) + 1); D[0, 1:] *= d
    D[1:, 0] = range(1, len(x) + 1); D[1:, 0] *= d

    for i in range(1, len(x)):
        for j in range(1, len(y)):
            cs = D[i-1, j-1] + sim(x[i], y[j])
            cd = D[i-1, j] + d
            ci = D[i, j-1] + d
            D[i,j] = max(cs, cd, ci)

    print(D)
    return D[len(x)][len(y)]
```

Keyboard-aware Substitutions?

- Compute cost of substitution by proximity of keys
- Map keys to grid, compute euclidean distance
- What about g <> h etc.?

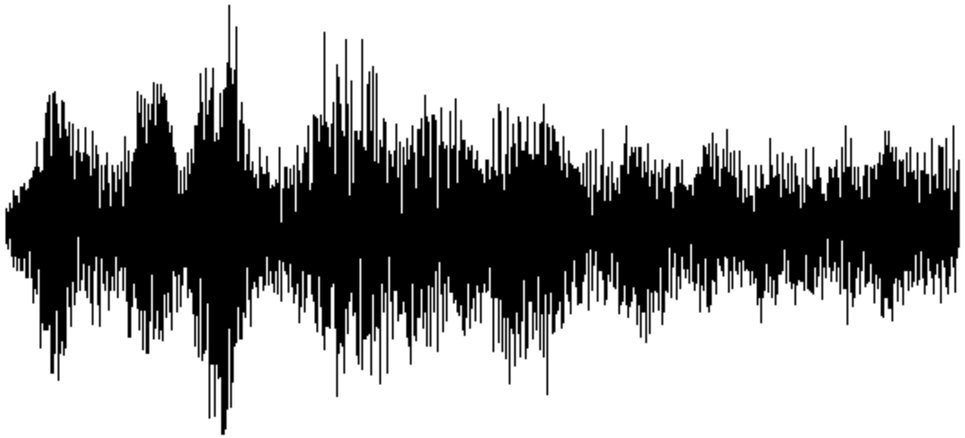
https://commons.wikimedia.org/wiki/File:KB_United_States.svg#/media/File:KB_United_States.svg

Outlook: Damerau-Levenshtein with adjacent transpositions

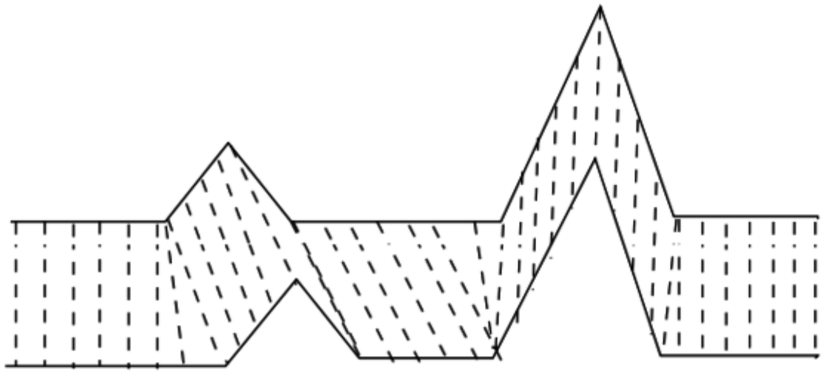
Comparing Sequences

…of non-discrete signals?

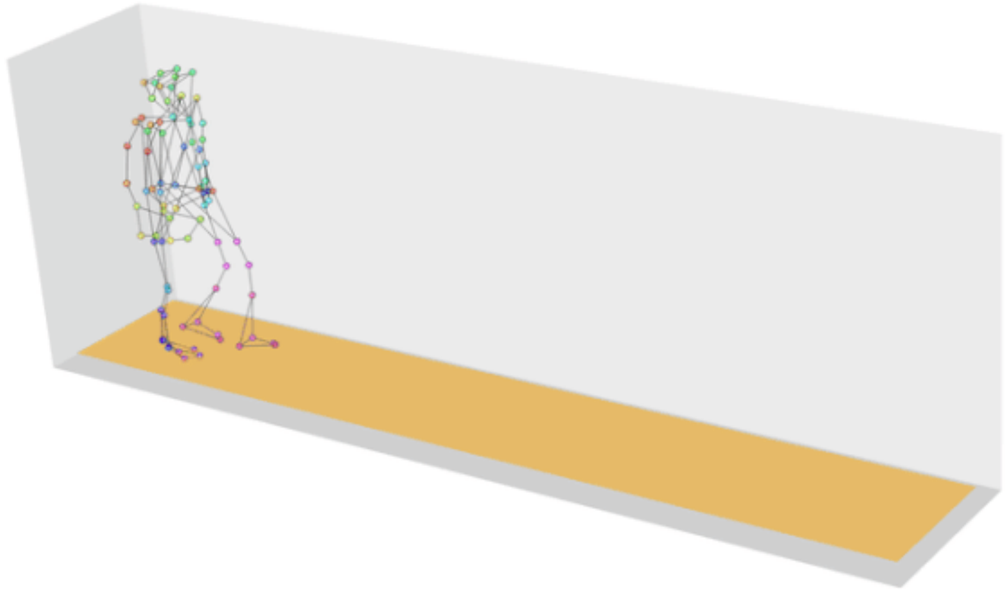
vs.



Dynamic Time Warping



https://en.wikipedia.org/wiki/File:Dynamic_time_warping.png



https://en.wikipedia.org/wiki/File:Two_repetitions_of_a_walking_sequence_of_an_individual_recorded_using_a_motion-capture_system.gif

Dynamic Time Warping

- Sequences assumed to be similar, no notion of insert
  —> first row/col is *inf*!
- Observations are continuous (not drawn from vocab)
  - There is *always* a cost
  - No explicit modeling of insertion/deletion/substitution

Dynamic Time Warping

```
def dtw(x: list, y: list, d) -> float:
    D = np.full((len(x) + 1, len(y) + 1), np.inf, dtype=float)
    D[0, 0] = 0

    for i in range(1, len(x)):
        for j in range(1, len(y)):
            cost = d(x[i], y[j])
            D[i, j] = cost + min(D[i-1, j],
                                 D[i, j-1],
                                 D[i-1, j-1])

    return D[len(x)][len(y)]
```

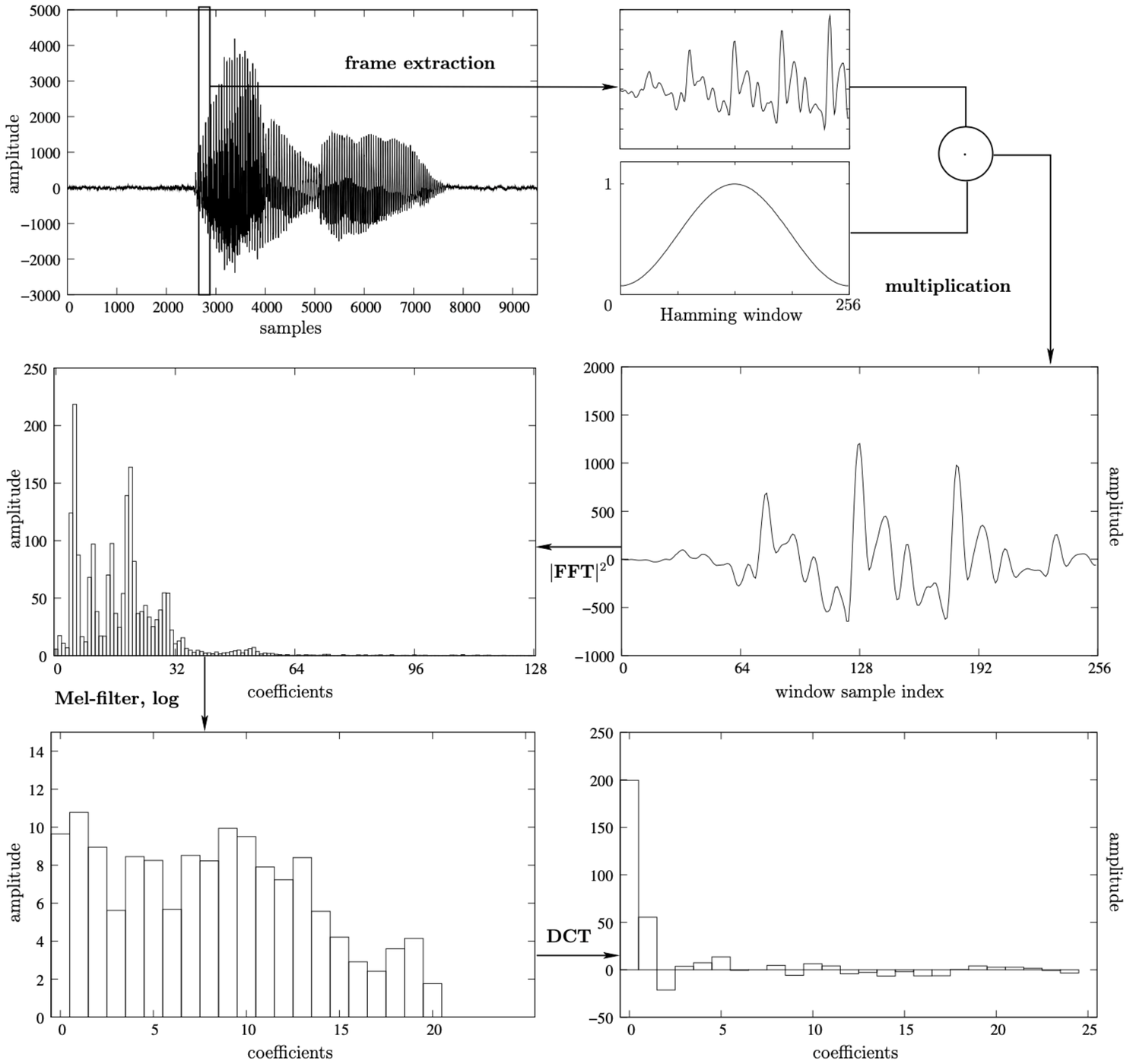
Dynamic Time Warping

On audio data

- Raw sample data is way to numerous
- Compute spectral (cepstral) features, eg. MFCC
- Use euclidean distance on feature vectors

MFCC

pipeline

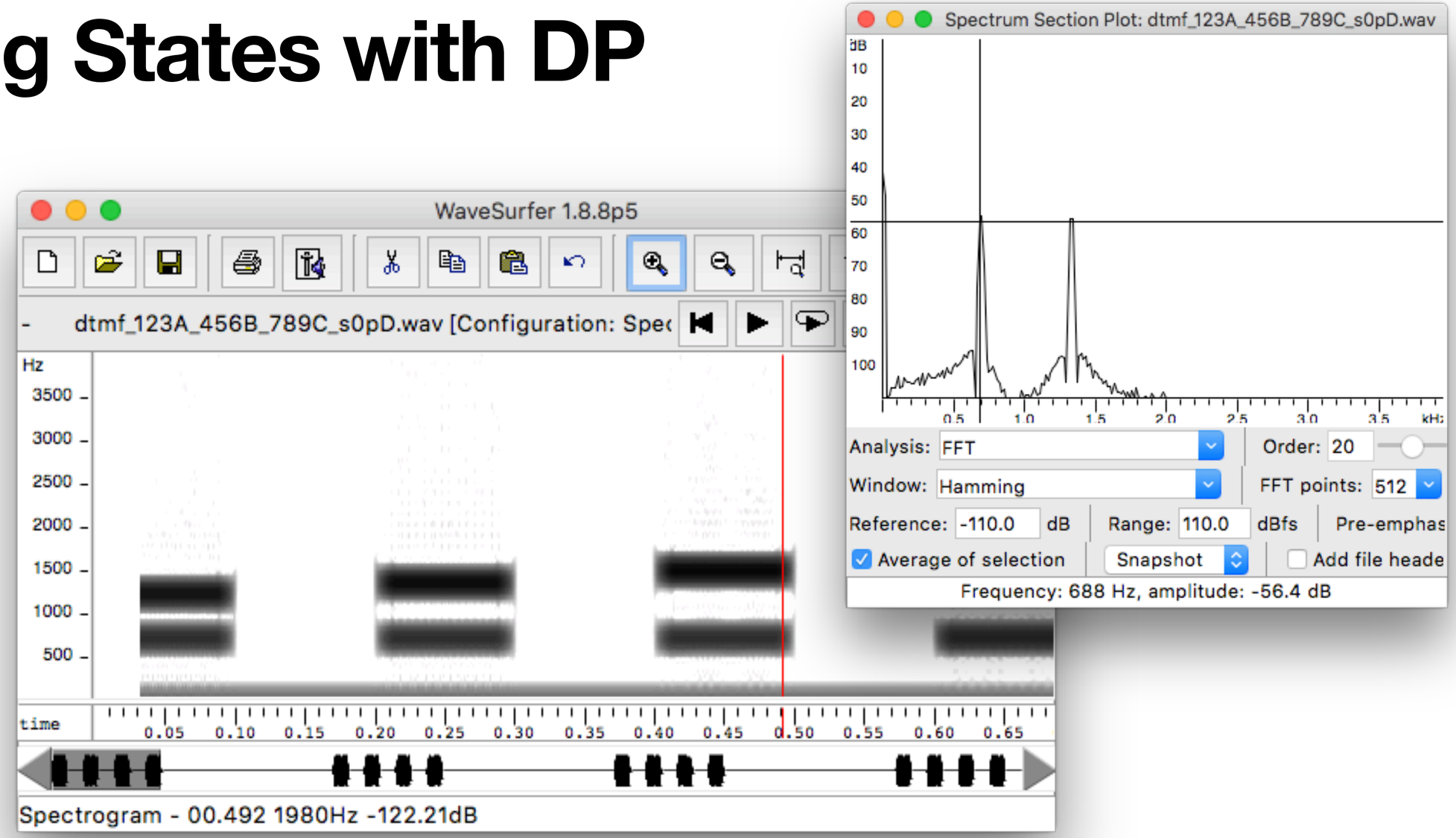


Multi-class Sequence Classification

Eg. Isolated word recognition

- Have (at least) one reference sequence per class (word)
- Compute DTW distance for test sequence to each reference
- Chose class with minimum distance
- How to speed up…?

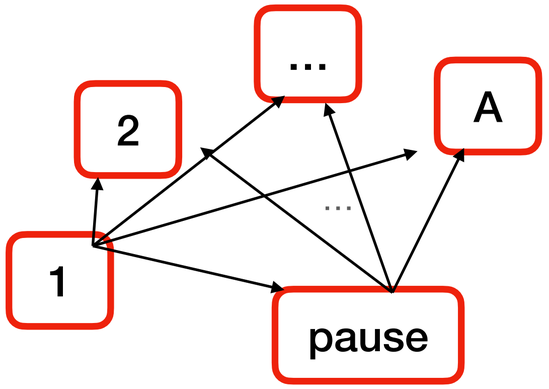
Decoding States with DP

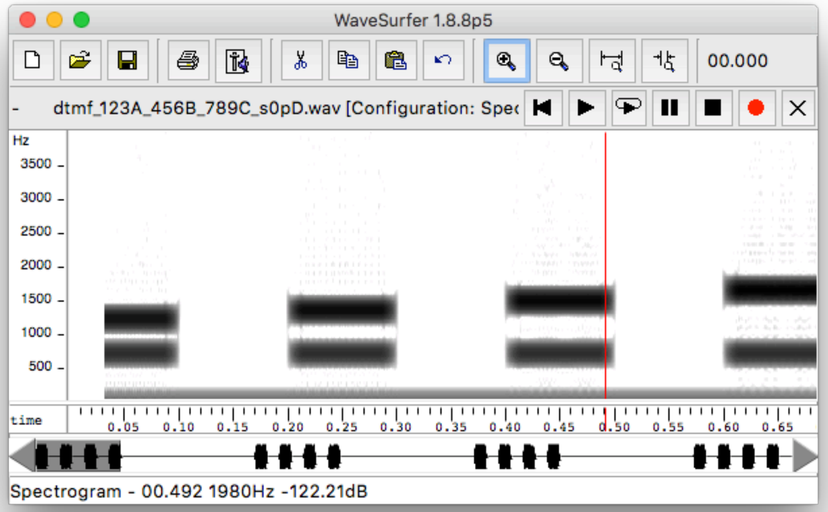


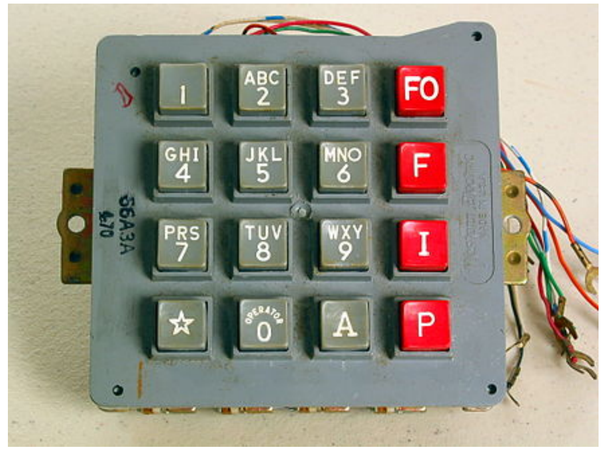
Dual-tone multi-frequency signaling (DTMF)

Decoding States

- Observation: *pause 1 pause 2 pause …*
- If we ignore noise etc.: 12+1 classes
  - Model each class as state
- DTMF sequence = path through graph



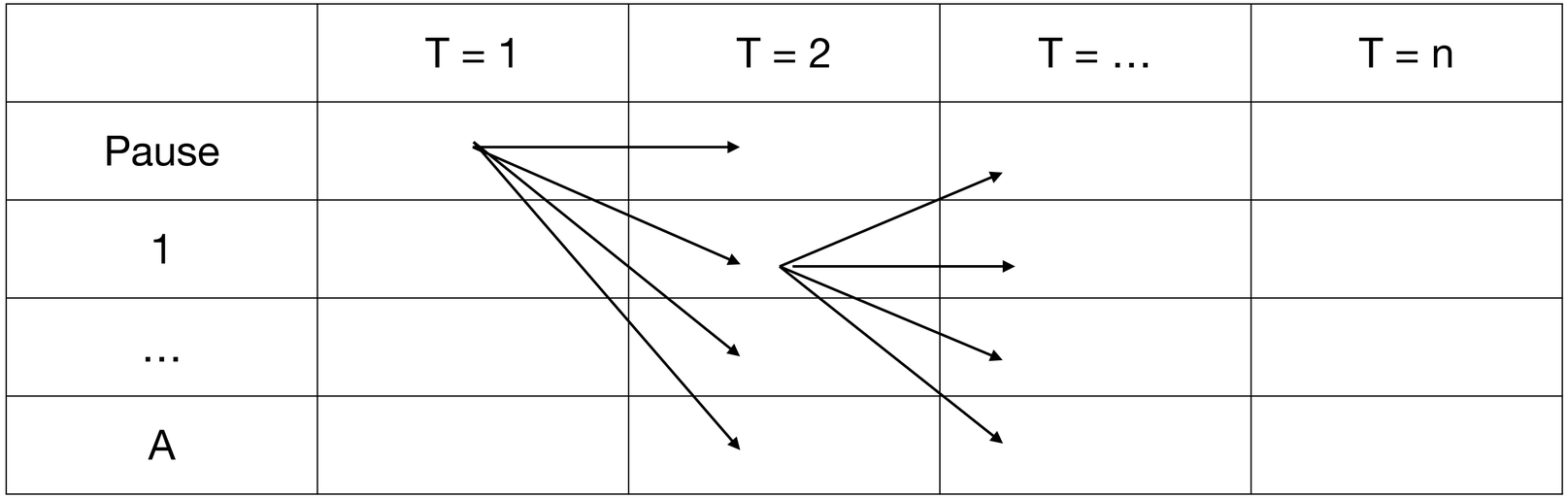




https://en.wikipedia.org/wiki/File:66a3aDTMFpad.jpg

Decoding States

- Protoypes for each state (data-driven? Explicitly modeled?)
- Could begin in any state? Or in *pause*?
- Left-to-right (time-synchronous)
- Any transition ok?
- Chose min of previous column



Assignment 1

Due April 11

- Edit distances (Hamming, Levenshtein, Needleman-Wunsch)
- Auto-Complete (using basic word stats)
- Isolated Word Recognition using DTW (on digits)
- DTMF sequence decoding using DP on states
- Submit via Moodle, Q&A on Teams

https://github.com/seqlrn/1-dynamic-programming