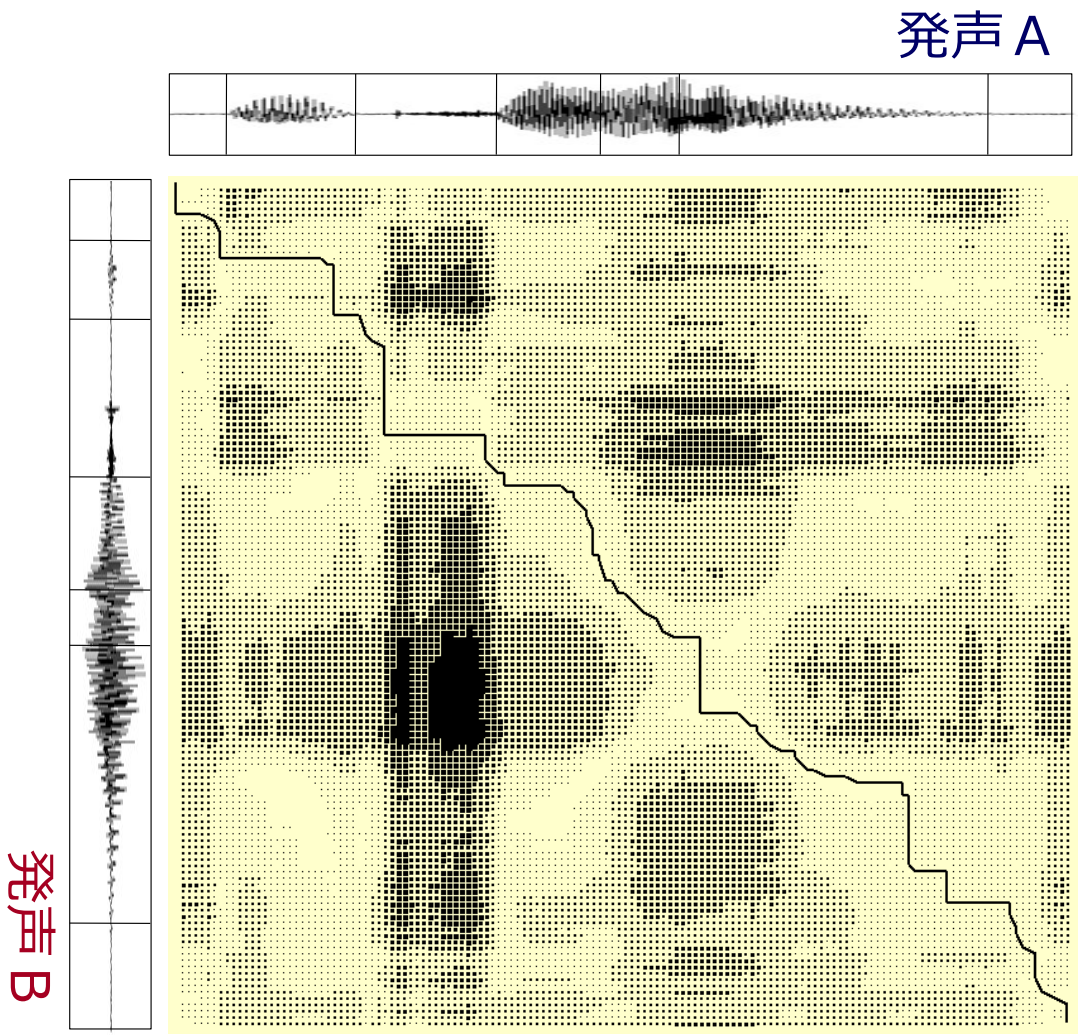
時間軸伸縮マッチング



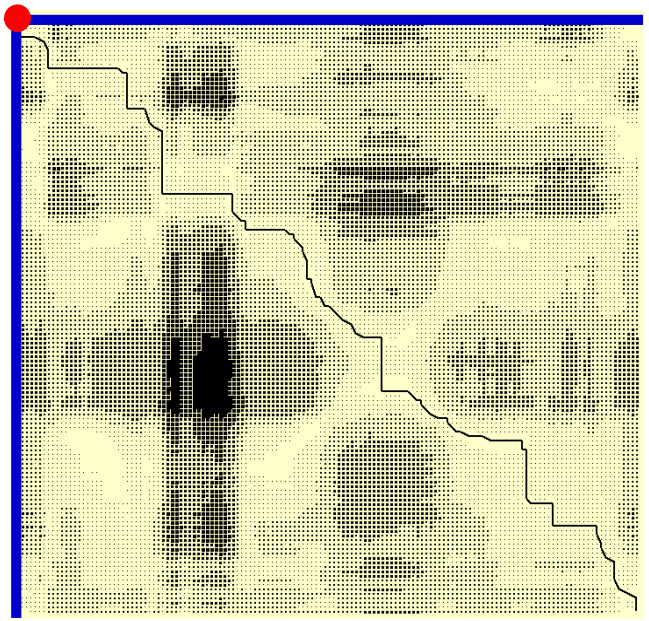
発声Aと発声Bの各分析フレーム
相互間の距離（局所距離）
→格子点の濃度で表現

累積距離が最短となる経路を
探索する問題に
動的計画法(DP)を適用

OKLAB@CIT

DPマッチングのアルゴリズム (1)

- 記号
  - ⇨正解（テンプレート）側のフレーム数　I
  - ⇨入力側のフレーム数　J
  - ⇨局所距離 $d(i,j)$
  - ⇨累積距離 $g(i,j)$



- 初期条件　（図の ● ）
  - ⇨$g(0,0) = d(0,0)$

- 境界条件　（図の ―― ）
  - ⇨$i > 0$ において $g(i,0) = g(i-1,0) + d(i,0)$
  - ⇨$j > 0$ において $g(0,j) = g(0,j-1) + d(0,j)$

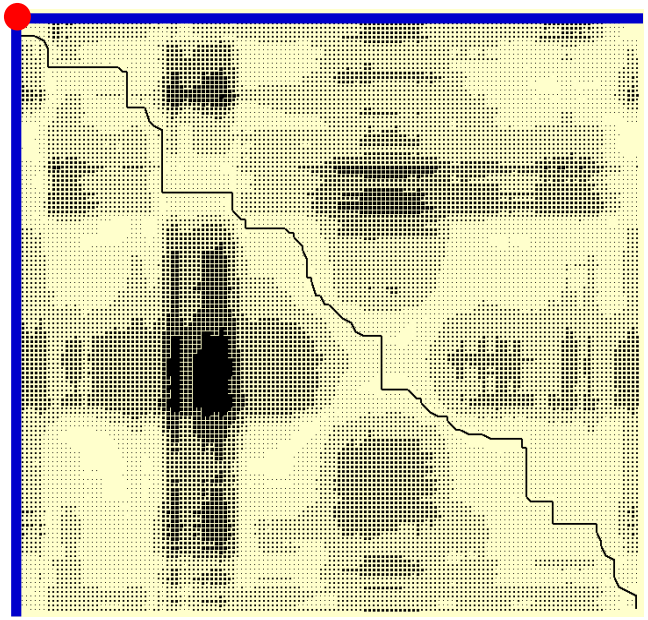
OKLAB@CIT

DPマッチングのアルゴリズム (2)

- その他のノード全て

$$g(i,j) = \min \begin{bmatrix} g(i,j-1) & + & d(i,j) \\ g(i-1,j-1) & + & 2d(i,j) \\ g(i-1,j) & + & d(i,j) \end{bmatrix}$$

  - ⇨再帰方程式（漸化式）
  - ⇨DPマッチングの標準形
  - ⇨別の遷移型も提案されている



- 最終処理
  - ⇨最終フレームでの累積距離 $g(I,J)$
  - ⇨フレーム数の和 $(I+J)$ で正規化

$$T = g(I,J) / (I+J)$$

OKLAB@CIT

DPマッチングのアルゴリズム(3)

- ex. 100単語の音声認識
  - ⇨テンプレートが100個（フレーム数 $I_n = I_1 \sim I_{100}$）
  - ⇨未知入力 （フレーム数 J）
  - ⇨100回のDPマッチングにより $T_n = T_1 \sim T_{100}$ を計算
  - ⇨ $\min T_n$ を与える n 番目の単語が認識結果

- 最適経路の意味
  - ⇨認識問題を解くだけなら、陽に最適経路を知る必要はない
  - ⇨最終フレーム (I, J) までの最適経路を与える累積距離が重要
  - ⇨音素ラベリング（音素境界にラベル付け）には有用

大レポート課題①　DPマッチングによる単語音声認識 (1)

▶ 概要

⇨ DPマッチングのアルゴリズムを利用し、小語彙の単語音声認識実験を行う

⇨ 音声入力～音響分析までの過程はすでに終了しているものとし、
予め用意されたテキストファイルのデータを利用する

⇨ 100単語のテンプレートに対して、
同じ発声内容の100単語（同一話者または別話者）を未知入力音声と
見立てて順に入力し、何単語が正しく認識できるか調べる

▶ データファイル

⇨ `city_mcepdata.zip` をダウンロード

⇨ 適当な場所に内容を展開

⇨ `city011, city012, city021, city022`の4つのフォルダ

⇨ 各フォルダにそれぞれ100個のテキストファイル

OKLAB@CIT

大レポート課題①　DPマッチングによる単語音声認識 (2)

- データの内容
  - ⇨100地名単語データベース
  - ⇨話者2名がそれぞれ2回ずつ発声（計4データセット、400単語分）
  - ⇨100単語の発声内容がすべて同じ順序で格納
  - ⇨ex. 先頭の単語（単語番号001）はAZABU（あざぶ）

- ファイル名
  - ⇨ex. `city011_001.txt`
  - ⇨`city011` は話者01の1回目発声（`city022`なら話者02の2回目発声）
  - ⇨`_001` は単語番号 （100単語なので`001`から`100`まで）

OKLAB@CIT

大レポート課題①　DPマッチングによる単語音声認識 (3)

- ファイル構造
  - ⇨テキスト形式； Windowsのメモ帳やUNIXのcatコマンドで見える
  - ⇨先頭に3行のヘッダ情報
    - 1行目：　ファイル名から拡張子 `.txt` を除いたテキスト (ex. `city011_001`)
    - 2行目：　発声内容（音素の略式表示）
    - 3行目：　フレーム数
  - ⇨4行目から後がデータ（音響特徴量ベクトル）
    - 1行が1フレームに対応
    - 15次のメルケプストラム特徴量　（空白で区切られた15個の浮動小数点数）
  - ⇨ex. フレーム数が61の場合、ファイルの行数は61 + 3（ヘッダ部）= 64行
  - ⇨単語によりフレーム数（発声時間長に対応）が異なるので行数も異なる

OKLAB@CIT

大レポート課題①　DPマッチングによる単語音声認識 (4)

- ▶ 実験方法
  - ⇨ 4データセットのうち2つを利用
  - ⇨ テンプレート（正解）1つ、未知入力（認識対象）1つの組み合わせ
  - ⇨ 同一話者（2通り）　　＝ 特定話者
    - ➧ ex. テンプレートが話者01の1回目、未知入力が話者01の2回目
  - ⇨ 別話者（4通り）　　≒不特定話者
    - ➧ ex. テンプレートが話者01の1回目、未知入力が話者02の1回目
  - ⇨ それぞれの組み合わせについて100単語×100単語の総当たり認識
  - ⇨ 発声内容はすべて同じ； 未知入力のN番目の単語をテンプレートの100単語とマッチングさせた結果、N番目の単語に対して最小の累積距離が得られれば正解
  - ⇨ 正解数（最小の累積距離を与えた単語が入力単語と一致した数）
    ＝ 単語認識率 [%]

OKLAB@CIT

大レポート課題①　プログラミングのヒント(1)

- データの読み込み
  - ⇨1単語のデータ構造は `double data[frame][dimension]`　（2次元）
  - ⇨この実験では `dimension` = 15で固定； `frame` は単語により異なる
  - ⇨テンプレートと未知入力それぞれ100単語を予めすべて読み込んでおく

- 局所距離の計算（パズルにおける○内の数字に相当）
  - ⇨テンプレートA（フレーム数 I ）と未知入力B（フレーム数 J ）のマッチング
  - ⇨Aのフレーム i のデータを a_{ik}、Bのフレーム j のデータを b_{jk} （k は次元）
  - ⇨ a_{ik} と b_{jk} の間の局所距離：

$$d(i,j) = \sqrt{(a_{i,1} - b_{j,1})^2 + (a_{i,2} - b_{j,2})^2 + \cdots + (a_{i,15} - b_{j,15})^2}$$

  - ⇨すべてのフレーム相互間について計算

OKLAB@CIT

大レポート課題①　プログラミングのヒント(2)

- DPマッチングのアルゴリズム
  - ⇨初期条件と境界条件
  - ⇨その他の格子点における累積距離 $g(i, j)$ を順次計算
  - ⇨最終点までの累積距離 $g(I, J)$
  - ⇨$T = g(I, J) / (I + J)$ … AとBの単語間距離

大レポート課題①　その他のヒントと考察事項

- 用いる単語数
  - ⇨システムの性能を知るには1～数単語だけ調べても意味なし
  - ⇨テンプレート、未知入力共に100単語
    → 100 × 100 ＝ 10000通りの組合せ

- バックトラック（経路を逆にたどる）
  - ⇨認識する（認識率を計算する）だけなら必要なし
  - ⇨DPの考察には適； 3Dグラフィック表示

- 斜め遷移の重み
  - ⇨局所距離を2倍 → 1倍や $\sqrt{2}$ 倍だとどうなる？

- 整合窓
  - ⇨対角線から離れたノードの計算を省略 → 性能にどう影響？

OKLAB@CIT