ゴガテイ
03179110

プログラミング基礎演習レポート２

1、導入:

人間が複数の音源から、必要な情報を抽出する機構を持っている。今回はその能力を模倣するICA（独立成分分析 ）のアルゴリズムを実装する。ICAにより、観察データ$\boldsymbol{x}$から、源信号$\mathbf{y}$が求められる。

２、手法＆結果:

問題１（必須）

①手法

１）まずはデータを読み、$\boldsymbol{x}$ に保存する、そして$\boldsymbol{x}$ と各データの数nを返すーーGetdata()

２）そして$\boldsymbol{x}$を調整し、$\boldsymbol{x} \leftarrow \boldsymbol{x} - \mathbb{E}[\boldsymbol{x}]$、また$\boldsymbol{\Sigma} = \mathbb{E}[\boldsymbol{x}\boldsymbol{x}^\top]$ により、$\boldsymbol{\Sigma}$(sigma)を求める。linalg.eigにより、$\boldsymbol{\Sigma}$の固有値dと固有ベクトルEが求められる。D = np.diag(1/d)により、$\boldsymbol{V} = \boldsymbol{E}\boldsymbol{D}^{-1/2}\boldsymbol{E}^\top$ 中のDを求め、さらに$\boldsymbol{V}$が計算できる。$\boldsymbol{z} = \boldsymbol{V}\boldsymbol{x}$により、$\boldsymbol{z}$がわかる。ーーWhitening ()

３）まずはWをランダムに生成する。大きさは（N,N）ここでは(2,2)、そして、かく$\boldsymbol{w}$ に対し、正規化する、また$\boldsymbol{w} \leftarrow \mathbb{E}[\boldsymbol{z}(\boldsymbol{w}^\top \boldsymbol{z})^3] - 3\boldsymbol{w}$ で$\boldsymbol{w}$を更新する、とともにWでも、こういう間、もちろん更新される。ここで、W <- (W*W.T)^(-0.5)*Wにより、Wを非相関にする（後の結果見ると、効果はそんなに著しいのではない）。Wが収束するまで、$\boldsymbol{w} \leftarrow \mathbb{E}[\boldsymbol{z}(\boldsymbol{w}^\top \boldsymbol{z})^3] - 3\boldsymbol{w}$ と$\boldsymbol{w}$の正規化を繰り返す。収束の条件はここで、abs(W * W.T-1)<0.0001にする。ーーGerneratew()&Decorrelation()

４）計算したWで、$\mathbf{y} = \mathrm{W}\boldsymbol{z}$で$\mathbf{y}$を求める。

５）ここでshow_data()を書いて、データの図を表示する。

②結果：結果の図はsource1.pngとsource2.pngに保存する。

Wを非相関化する処理を一応したけど、あまり著しくないため、ランダムのWは適切に決めて、kadai1.ipynbを完成した。

ここでW = [[0.7335677 0.45488916][0.48046912 0.4304221]]

ICA問題はそもそも解がいくつあるから、信号源の幅はちょっと変わった。

問題２（自由）

①手法

ここの処理の手法と前は完全に同じであるから、ただデータ処理か出力か書く

ここではファイルが大きから、＃で省略したものが多い、もし先生がチエックしたいなら、＃を削除してお願いします。そして、music1.wavとmusic2.wavの添付をも、お願いします。

１）ここではpyaudioを使い、音声を実時的に流そうとする。なぜかというと、元々はscipy.io.wavfile.read とscipy.io.wavfile.write を使ったが、出力はいつもおかしくなってしまう。色々調べて、pyaudioにした。後でyはcomplex128ということもわかり、ちょっと圧縮し、wavをも生成できる。だが、やはり実時のデータは圧縮前だから、wavより美しいと思うから（実時出力には#削除必要あり）

②結果：

ここで、二つのWを選び、それぞれのWはsource1とsource2をよくする。

W=[[0.95180884 0.55365452] [0.11275333 0.10250868]]は Eliceをよく分離できる。

W=[[0.85359471 0.88491861] [0.38882422 0.10559617]]はMozartをよく分離できる。

ipynbを実行したら、対応する、分離音声がsep1,sep2として、出力できる。（#削除必要あり）、ここでは便利のため、よく分離された音声をsource1.wavとsource2.wavに保存する。

問題 3 （自由）

①手法

１）ここでも、ただデータ処理か出力か書く、mpimg.imreadで、元々の画像を読み、表示する。そして、サイズを変えて、後の処理を行う。かく画像を(512*512,1)にする。imsaveで、画像を保存する。

結果

ここでも、二つのWを選び、それぞれのWはsource1とsource2をよくする。

W=[[0.7768532 0.38323983] [0.21723714 0.66516969]]は Lenaをよく分離できる。

W=[[0.98194907 0.47126819] [0.65710421 0.79107614]]はnot Lena(another woman)をよく分離できる。

それぞれのipynbを実行したら、対応する、分離画像がp1,p2として、出力できる。だが、ここでは便利のため、よく分離された画像をsource1.pngとsource2.pngに保存する。

問題 4 （自由）

ここは考えだけがある、ある重なる画像（手振れなどの原因で）二つの画像を処理すれば、いいと思う。

3 、考察:

一週間ぐらいやりました。最初は行列の計算で、どれがNか、どれがnか、色々悩んだことがある。ようやく問題 1 がとけたけど、問題 2 を解いている間、20秒の音声は 2 分間の騒音になってしまった。正直、がっかりしていた、問題はどこにあるか色々排除した。そして色々調べて、実時的に音声を出力する方法を見つけ、やってみると、なんと、綺麗なEliceが分離された。心から喜んでいる。もちろんLENAの画像分離も面白いけど、言葉だけでは表示できない喜びはEliceのものである。

4、参考:

• Hyvärinen, Aapo, Juha Karhunen, and Erkki Oja. Independent component analysis. Vol. 46. John Wiley & Sons, 2004