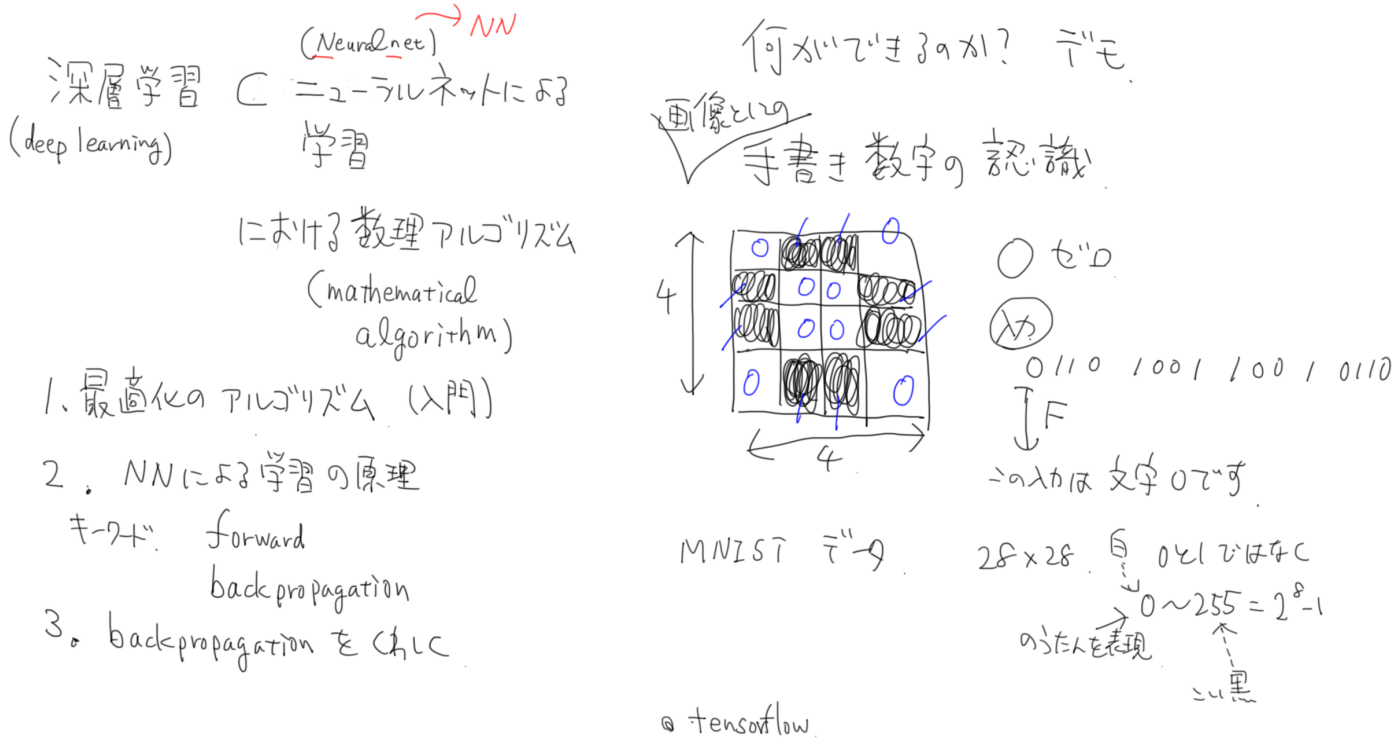
Movie in the mp4 format.
動画: Neural Network とは? (youtube, 限定公開).
プログラム (for pytorch on python)
手書き数字の認識: pytorch で MNIST (動画で紹介しているページ)

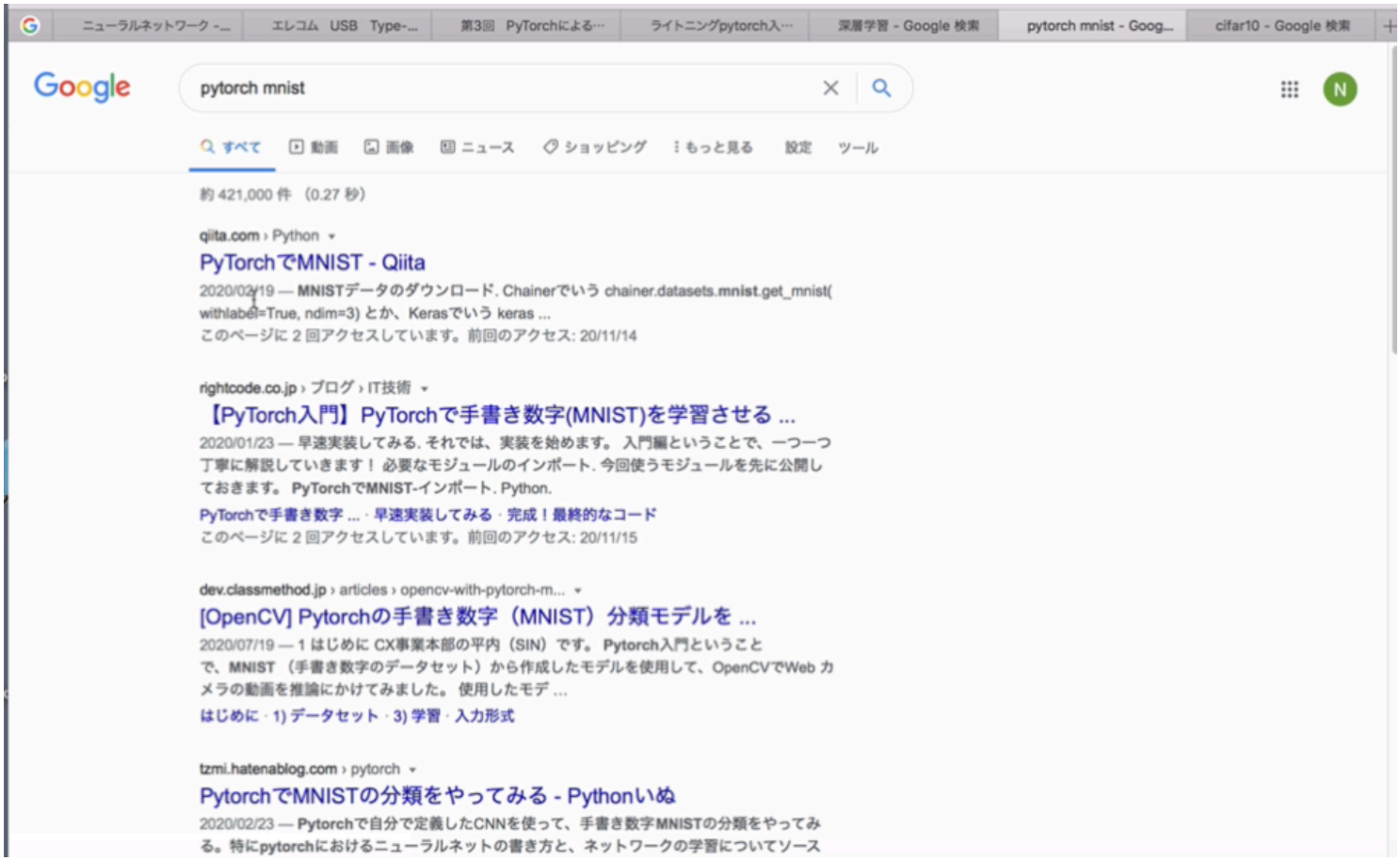
0050.png



0100.png


深層学習
ニューラルネットワーク -...
エレコム USB Type-...
第3回 PyTorchによる…
ライトニングpytorch入…
深層学習 - Google 検索
pytorch mnist - Goog...
cifar10 - Google 検索
Google
深層学習
すべて
ニュース
画像
ショッピング
動画
もっと見る
設定
ツール
約 4,660,000 件 （0.53 秒）
ディープラーニング(深層学習)とは、人間が自然に行うタスクをコンピュータに学習させる機械学習の手法のひとつです。 人工知能(AI)の急速な発展を支える技術であり、その進歩により様々な分野への実用化が進んでいます。
jp.mathworks.com › discovery › deep-learning
ディープラーニング – これだけは知っておきたい3つのこと ...
強調スニペットについて
フィードバック
ledge.ai › Topic Keywords › AI:人工知能
ディープラーニング（深層学習）とは｜基本知識・仕組み ...
2020/02/10 — ディープラーニング（Deep Learning）または深層学習とは、ニューラルネットワークを多層に結合し表現・学習能力を高めた機械学習の手法です。歴史から仕組み、人工知能（AI）、ニューラルネットワークや機械学習との ...
ja.wikipedia.org › wiki › ディープラーニング
ディープラーニング - Wikipedia
その他の画像
ディープラーニング
ディープラーニングまたは深層学習とは、多層の人工ニューラルネットワークによる機械学習手法である。要素技術としてはバックプロパゲーションなど、20世紀のうちに開発されていたものの、4層以上の深層ニューラルネットについて、局所最適解や勾配消失などの技術的な問題によって十分学習させられず、性能も芳しくなかった。 ウィキペディア
他の人はこちらも検索
他 15 件以上を表示
機械学習
人工知能
Python
ロボット工学
フィードバック

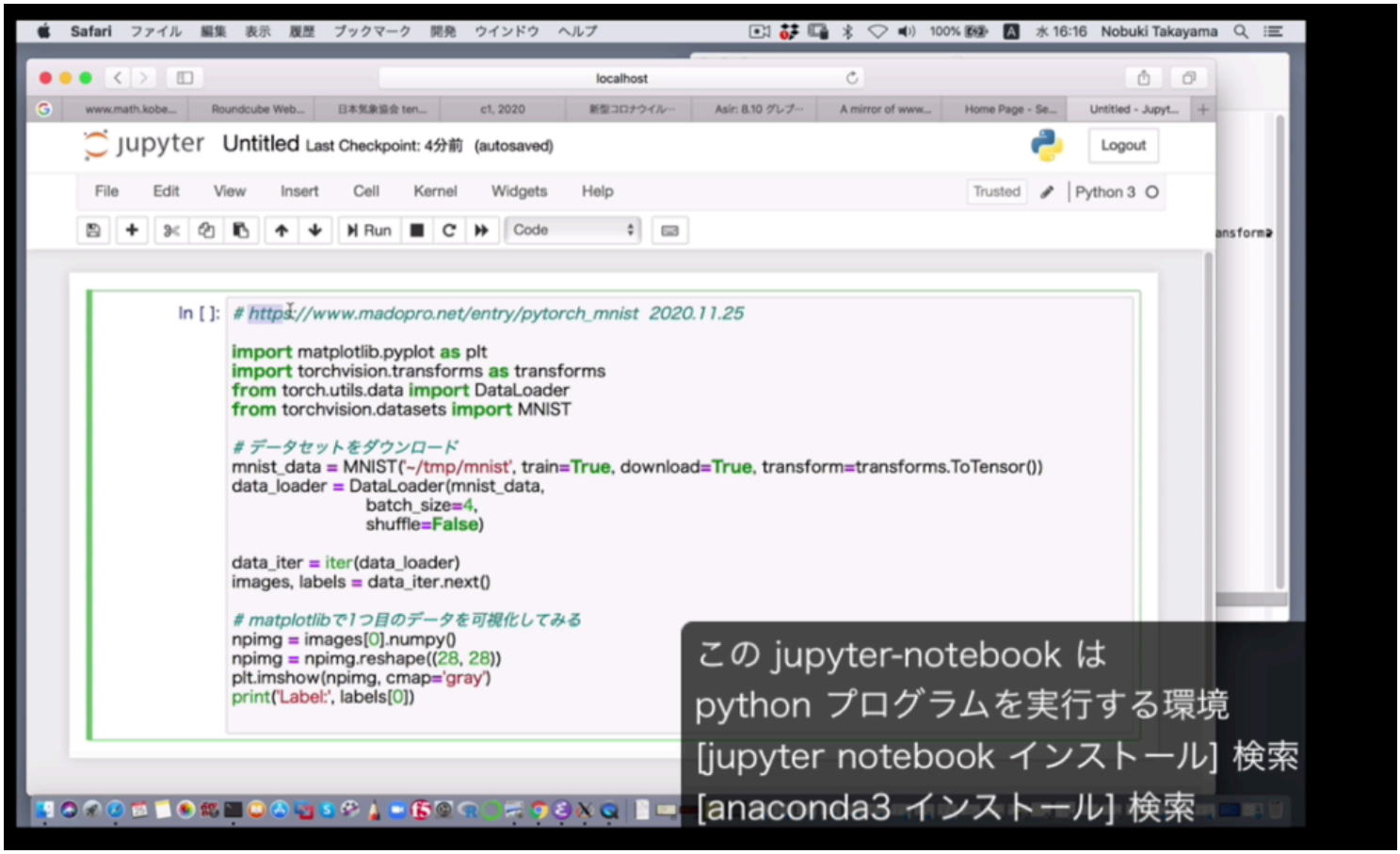
0200.png


ニューラルネットワーク -...
エレコム USB Type-...
第3回 PyTorchによる…
ライトニングpytorch入…
深層学習 - Google 検索
pytorch mnist - Goog...
cifar10 - Google 検索
Google
pytorch mnist
すべて
動画
画像
ニュース
ショッピング
もっと見る
設定
ツール
約 421,000 件 （0.27 秒）
qiita.com › Python
PyTorchでMNIST - Qiita
2020/02/19 — MNISTデータのダウンロード. Chainerでいう chainer.datasets.mnist.get_mnist(withlabel=True, ndim=3) とか、Kerasでいう keras ...
このページに 2 回アクセスしています。前回のアクセス: 20/11/14
rightcode.co.jp › ブログ › IT技術
【PyTorch入門】PyTorchで手書き数字(MNIST)を学習させる ...
2020/01/23 — 早速実装してみる. それでは、実装を始めます。 入門編ということで、一つ一つ丁寧に解説していきます！ 必要なモジュールのインポート. 今回使うモジュールを先に公開しておきます。 PyTorchでMNIST-インポート. Python.
PyTorchで手書き数字 ... · 早速実装してみる · 完成！最終的なコード
このページに 2 回アクセスしています。前回のアクセス: 20/11/15
dev.classmethod.jp › articles › opencv-with-pytorch-m...
[OpenCV] Pytorchの手書き数字（MNIST）分類モデルを ...
2020/07/19 — 1 はじめに CX事業本部の平内（SIN）です。 Pytorch入門ということで、MNIST （手書き数字のデータセット）から作成したモデルを使用して、OpenCVでWeb カメラの動画を推論にかけてみました。 使用したモデ ...
はじめに · 1) データセット · 3) 学習 · 入力形式
tzmi.hatenablog.com › pytorch
PytorchでMNISTの分類をやってみる - Pythonいぬ
2020/02/23 — Pytorchで自分で定義したCNNを使って、手書き数字MNISTの分類をやってみる。特にpytorchにおけるニューラルネットの書き方と、ネットワークの学習についてソース

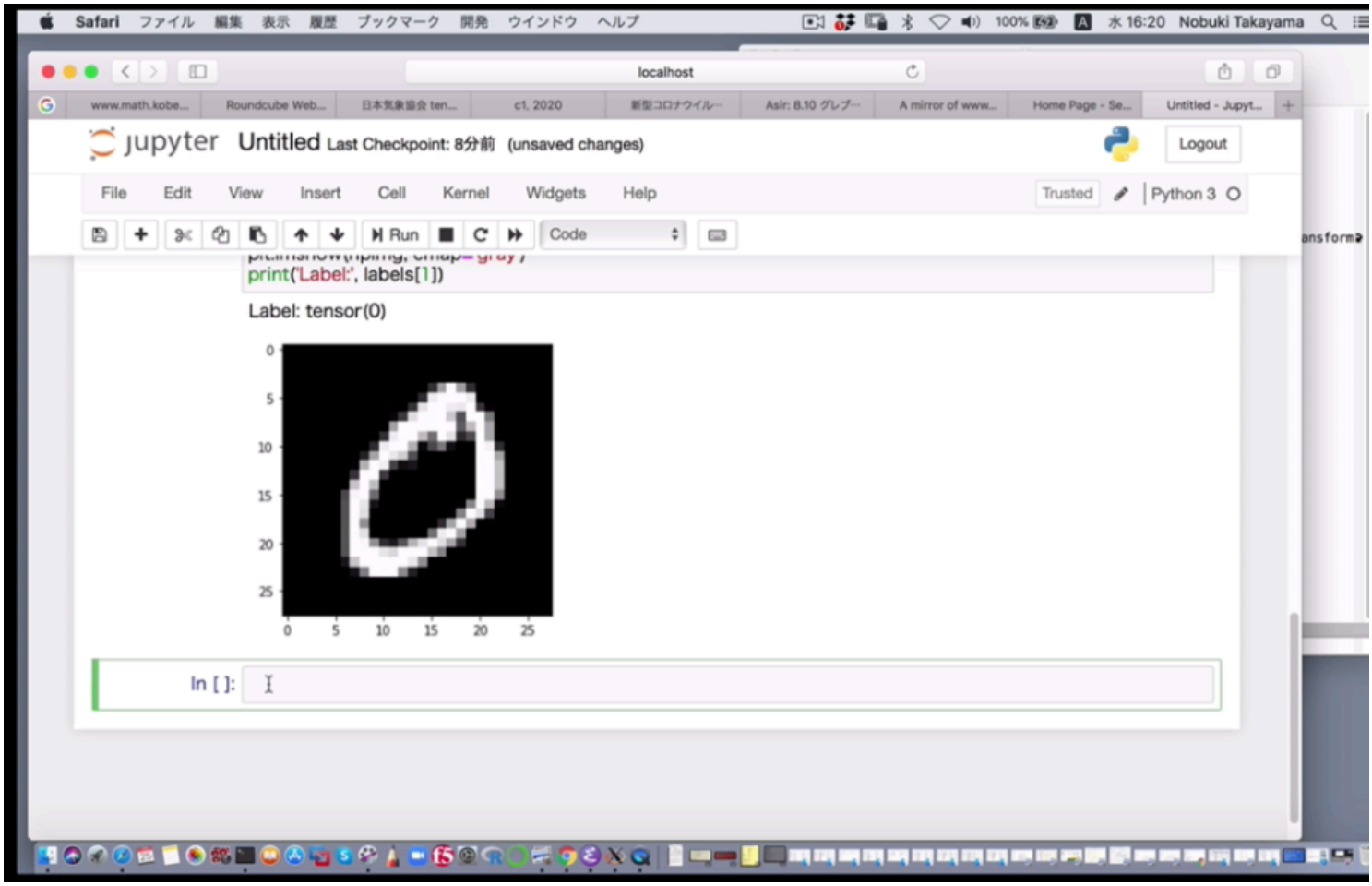
0300.png


cifar10
ニューラルネットワーク -...
エレコム USB Type-...
第3回 PyTorchによる...
ライトニングpytorch入...
深層学習 - Google 検索
pytorch mnist - Goog...
cifar10 - Google 検索
Google
cifar10
すべて
画像
動画
ニュース
書籍
もっと見る
設定
ツール
約 873,000 件 (0.38 秒)
www.atmarkit.co.jp › ait › articles › 2006/10 › news021
CIFAR-10：物体カラー写真（乗り物や動物など）の画像 ... - IT
2020/06/10 — データセット「CIFAR-10」について説明。6万枚の物体カラー写真（乗り物や動物など）の「画像＋ラベル」データが無料でダウンロードでき、画像認識などのディープラーニングに利用 ...
20/11/14 にこのページにアクセスしました。
newtechnologylifestyle.net › AIプログラミング
CIFAR-10のデータセットを用いてCNNの画像認識を行ってみる
2018/05/18 — CIFAR-10はKerasのデータセットに用意されているので、簡単にインポートして実行することが出来ます。 使用するデータ. 使用データ：CIFAR-10 訓練用データ：50000 テストデータ：10000 説明変数：画像 ...
aidiary.hatenablog.com › entry
CIFAR-10 - 人工知能に関する断創録
2015/10/14 — MNISTの数字画像はそろそろ飽きてきた（笑）ので一般物体認識のベンチマークとしてよく使われているCIFAR-10という画像データセットについて調べていた。 このデータは、約8000万枚の画像がある80 Million Tiny ...
uchidama.hatenablog.com › entry › 2018/01/11
KerasデータセットのCIFAR-10画像とラベルを表示する簡単 ...
2018/01/11 — CIFAR-10サンプルの学習は回して見ても、データの中身はちゃんと見てなかったので作って見ました。 jupyter notebookを使用して作りました。 CIFAR-10とは 一般物体認識のベンチマークとしてよく使われている画像 ...
https://www.atmarkit.co.jp/ait/articles/2006/10/news021.html

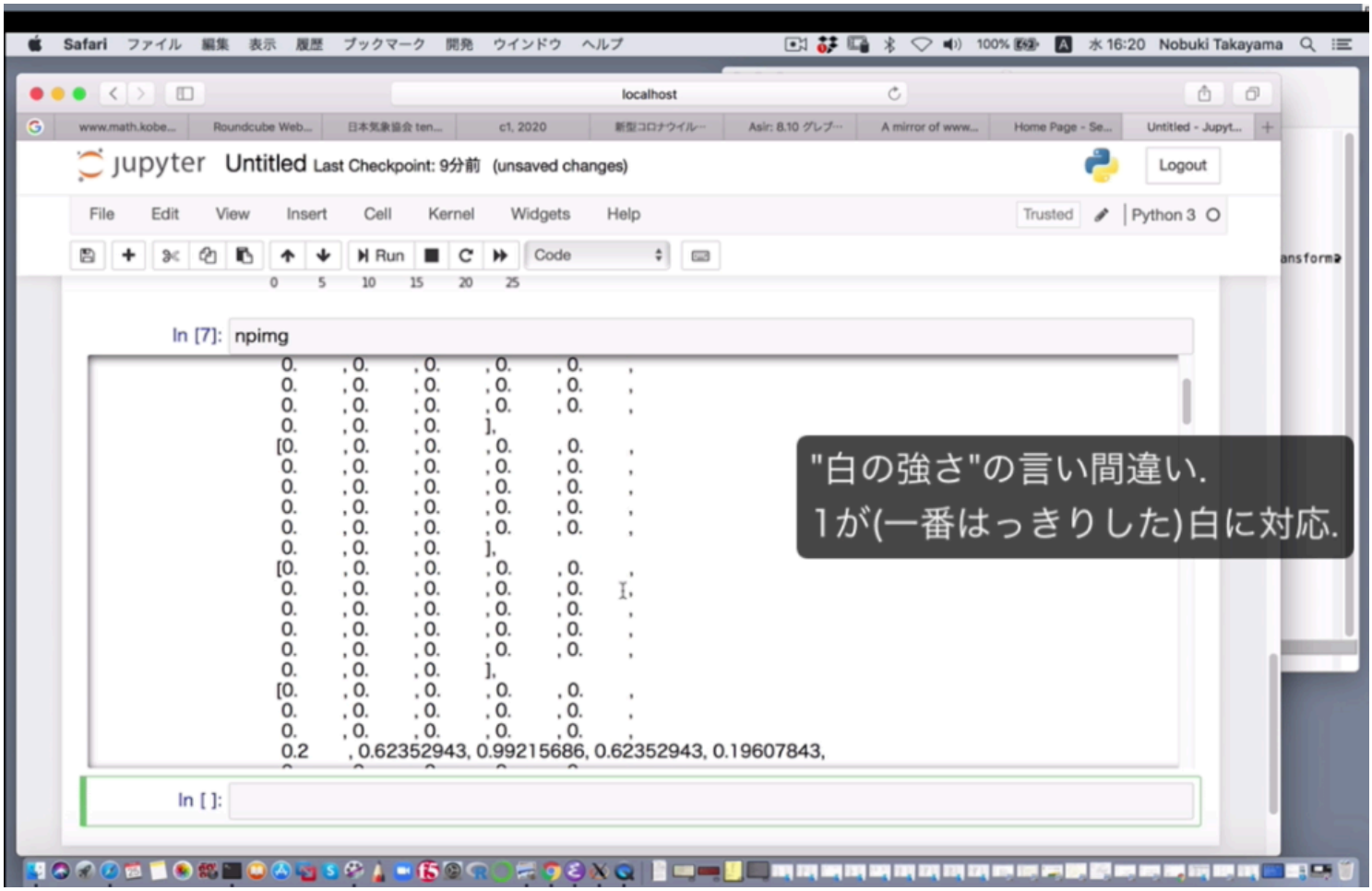
0400.png


jupyter Untitled Last Checkpoint: 4分前 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted Python 3
Run Code
In [ ]:
# https://www.madopro.net/entry/pytorch_mnist 2020.11.25
import matplotlib.pyplot as plt
import torchvision.transforms as transforms
from torch.utils.data import DataLoader
from torchvision.datasets import MNIST
# データセットをダウンロード
mnist_data = MNIST('~/tmp/mnist', train=True, download=True, transform=transforms.ToTensor())
data_loader = DataLoader(mnist_data,
batch_size=4,
shuffle=False)
data_iter = iter(data_loader)
images, labels = data_iter.next()
# matplotlibで1つ目のデータを可視化してみる
npimg = images[0].numpy()
npimg = npimg.reshape((28, 28))
plt.imshow(npimg, cmap='gray')
print('Label:', labels[0])
この jupyter-notebook は
python プログラムを実行する環境
[jupyter notebook インストール] 検索
[anaconda3 インストール] 検索

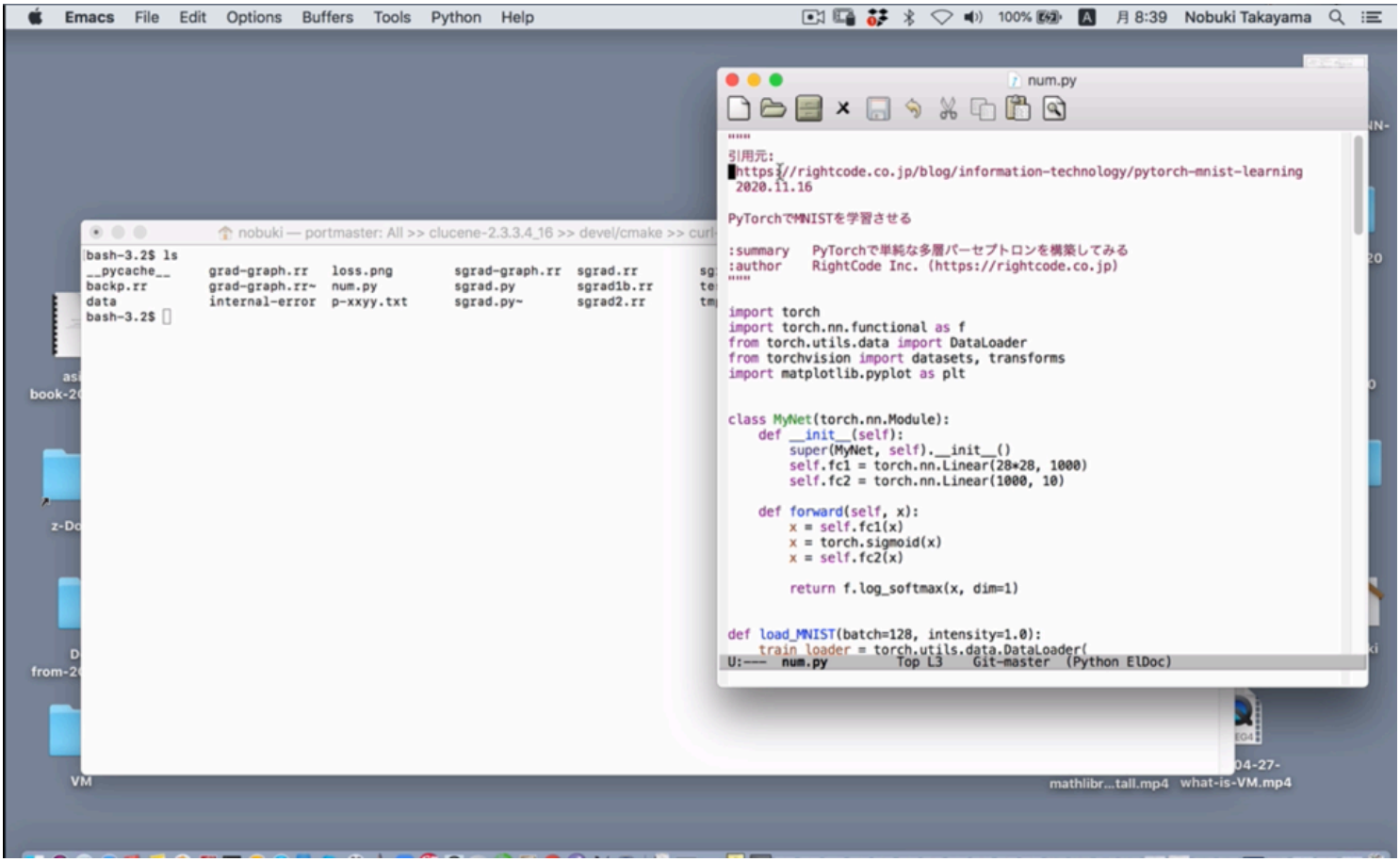
0420.png


Safari ファイル 編集 表示 履歴 ブックマーク 開発 ウインドウ ヘルプ
100% 水 16:20 Nobuki Takayama
localhost
jupyter Untitled Last Checkpoint: 8分前 (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted Python 3
Run Code
print('Label:', labels[1])
Label: tensor(0)
In [ ]:

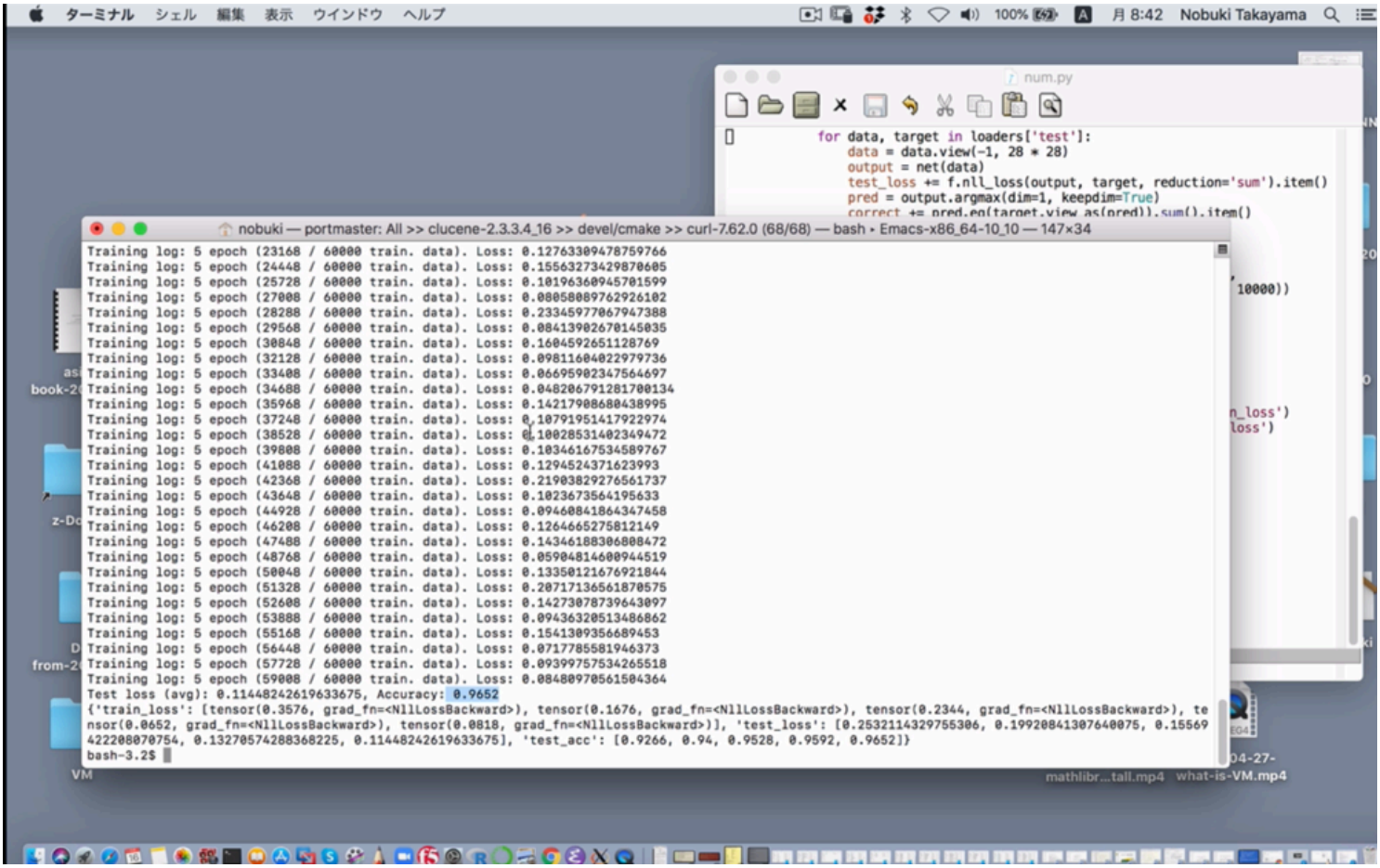
0440.png


Safari ファイル 編集 表示 履歴 ブックマーク 開発 ウインドウ ヘルプ
Nobuki Takayama
localhost
jupyter Untitled Last Checkpoint: 9分前 (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code
In [7]: npimg
0. , 0. , 0. , 0. , 0. ,
0. , 0. , 0. , 0. , 0. ,
0. , 0. , 0. , 0. , 0. ,
0. , 0. , 0.],
[0. , 0. , 0. , 0. , 0. ,
0. , 0. , 0. , 0. , 0. ,
0. , 0. , 0. , 0. , 0. ,
0. , 0. , 0. , 0. , 0. ,
0. , 0. , 0. , 0. , 0. ,
0. , 0. , 0.],
[0. , 0. , 0. , 0. , 0. ,
0. , 0. , 0. , 0. , 0.],
0. , 0. , 0. , 0. , 0. ,
0. , 0. , 0. , 0. , 0. ,
0. , 0. , 0. , 0. , 0. ,
0. , 0. , 0.],
[0. , 0. , 0. , 0. , 0. ,
0. , 0. , 0. , 0. , 0. ,
0. , 0. , 0. , 0. , 0. ,
0.2 , 0.62352943, 0.99215686, 0.62352943, 0.19607843,
In [ ]:
"白の強さ"の言い間違い.
1が(一番はっきりした)白に対応.

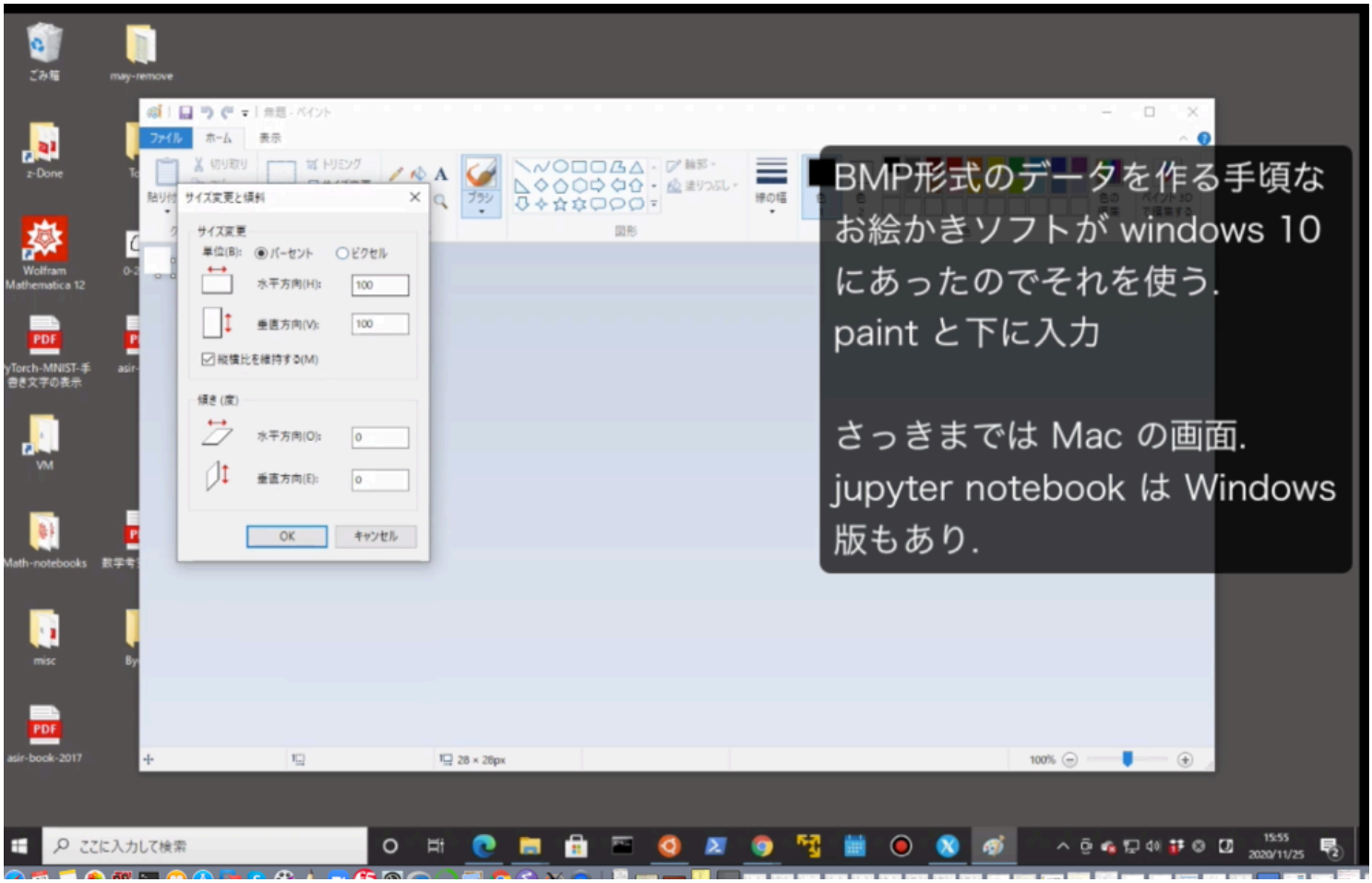
0600.png


Emacs File Edit Options Buffers Tools Python Help
100% 月 8:39 Nobuki Takayama
num.py
引用元:
https://rightcode.co.jp/blog/information-technology/pytorch-mnist-learning
2020.11.16
PyTorchでMNISTを学習させる
:summary PyTorchで単純な多層パーセプトロンを構築してみる
:author RightCode Inc. (https://rightcode.co.jp)
import torch
import torch.nn.functional as f
from torch.utils.data import DataLoader
from torchvision import datasets, transforms
import matplotlib.pyplot as plt
class MyNet(torch.nn.Module):
def __init__(self):
super(MyNet, self).__init__()
self.fc1 = torch.nn.Linear(28*28, 1000)
self.fc2 = torch.nn.Linear(1000, 10)
def forward(self, x):
x = self.fc1(x)
x = torch.sigmoid(x)
x = self.fc2(x)
return f.log_softmax(x, dim=1)
def load_MNIST(batch=128, intensity=1.0):
train_loader = torch.utils.data.DataLoader(
U:--- num.py Top L3 Git-master (Python ElDoc)
nobuki — portmaster: All >> clucene-2.3.3.4_16 >> devel/cmake >> curl
bash-3.2$ ls
__pycache__ grad-graph.rr loss.png sgrad-graph.rr sgrad.rr
backp.rr grad-graph.rr~ num.py sgrad.py sgrad1b.rr
data internal-error p-xxyy.txt sgrad.py~ sgrad2.rr
bash-3.2$
VM
mathlibr...tall.mp4
what-is-VM.mp4

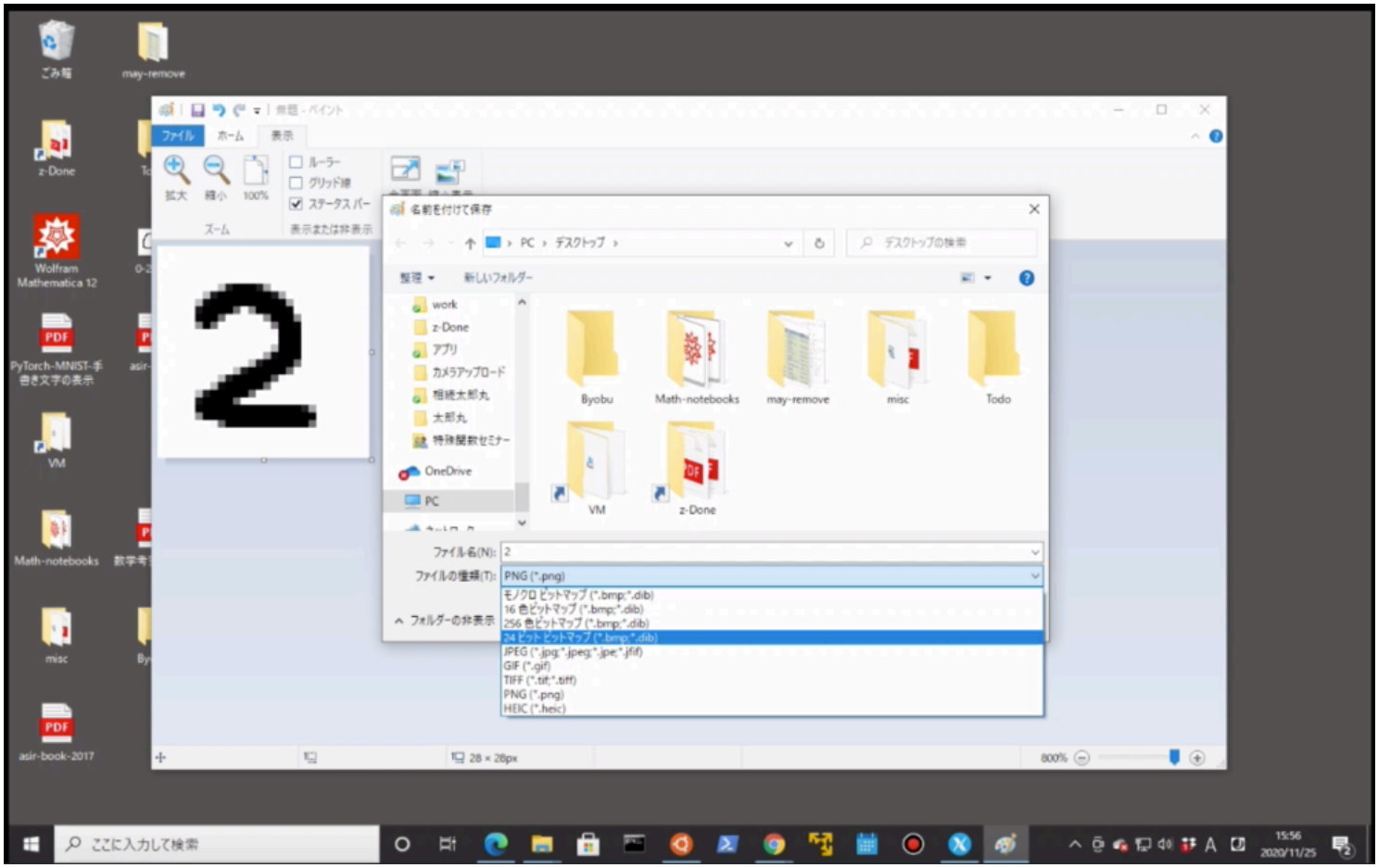
0700.png


num.py
for data, target in loaders['test']:
data = data.view(-1, 28 * 28)
output = net(data)
test_loss += f.nll_loss(output, target, reduction='sum').item()
pred = output.argmax(dim=1, keepdim=True)
nobuki — portmaster: All >> clucene-2.3.3.4_16 >> devel/cmake >> curl-7.62.0 (68/68) — bash • Emacs-x86_64-10_10 — 147×34
Training log: 5 epoch (23168 / 60000 train. data). Loss: 0.12763309478759766
Training log: 5 epoch (24448 / 60000 train. data). Loss: 0.15563273429870605
Training log: 5 epoch (25728 / 60000 train. data). Loss: 0.10196360945701599
Training log: 5 epoch (27008 / 60000 train. data). Loss: 0.08058089762926102
Training log: 5 epoch (28288 / 60000 train. data). Loss: 0.23345977067947388
Training log: 5 epoch (29568 / 60000 train. data). Loss: 0.08413902670145035
Training log: 5 epoch (30848 / 60000 train. data). Loss: 0.1604592651128769
Training log: 5 epoch (32128 / 60000 train. data). Loss: 0.09811604022979736
Training log: 5 epoch (33408 / 60000 train. data). Loss: 0.06695902347564697
Training log: 5 epoch (34688 / 60000 train. data). Loss: 0.048206791281700134
Training log: 5 epoch (35968 / 60000 train. data). Loss: 0.14217908680438995
Training log: 5 epoch (37248 / 60000 train. data). Loss: 0.10791951417922974
Training log: 5 epoch (38528 / 60000 train. data). Loss: 0.10028531402349472
Training log: 5 epoch (39808 / 60000 train. data). Loss: 0.10346167534589767
Training log: 5 epoch (41088 / 60000 train. data). Loss: 0.1294524371623993
Training log: 5 epoch (42368 / 60000 train. data). Loss: 0.21903829276561737
Training log: 5 epoch (43648 / 60000 train. data). Loss: 0.1023673564195633
Training log: 5 epoch (44928 / 60000 train. data). Loss: 0.09460841864347458
Training log: 5 epoch (46208 / 60000 train. data). Loss: 0.1264665275812149
Training log: 5 epoch (47488 / 60000 train. data). Loss: 0.14346188306808472
Training log: 5 epoch (48768 / 60000 train. data). Loss: 0.05904814600944519
Training log: 5 epoch (50048 / 60000 train. data). Loss: 0.13350121676921844
Training log: 5 epoch (51328 / 60000 train. data). Loss: 0.20717136561870575
Training log: 5 epoch (52608 / 60000 train. data). Loss: 0.14273078739643097
Training log: 5 epoch (53888 / 60000 train. data). Loss: 0.09436320513486862
Training log: 5 epoch (55168 / 60000 train. data). Loss: 0.1541309356689453
Training log: 5 epoch (56448 / 60000 train. data). Loss: 0.0717785581946373
Training log: 5 epoch (57728 / 60000 train. data). Loss: 0.09399757534265518
Training log: 5 epoch (59008 / 60000 train. data). Loss: 0.08488970561504364
Test loss (avg): 0.11448242619633675, Accuracy: 0.9652
{'train_loss': [tensor(0.3576, grad_fn=<NllLossBackward>), tensor(0.1676, grad_fn=<NllLossBackward>), tensor(0.2344, grad_fn=<NllLossBackward>), tensor(0.0652, grad_fn=<NllLossBackward>), tensor(0.0818, grad_fn=<NllLossBackward>)], 'test_loss': [0.2532114329755306, 0.19920841307640075, 0.15569422208070754, 0.13270574288368225, 0.11448242619633675], 'test_acc': [0.9266, 0.94, 0.9528, 0.9592, 0.9652]}
bash-3.2$
mathlibr...tall.mp4
what-is-VM.mp4

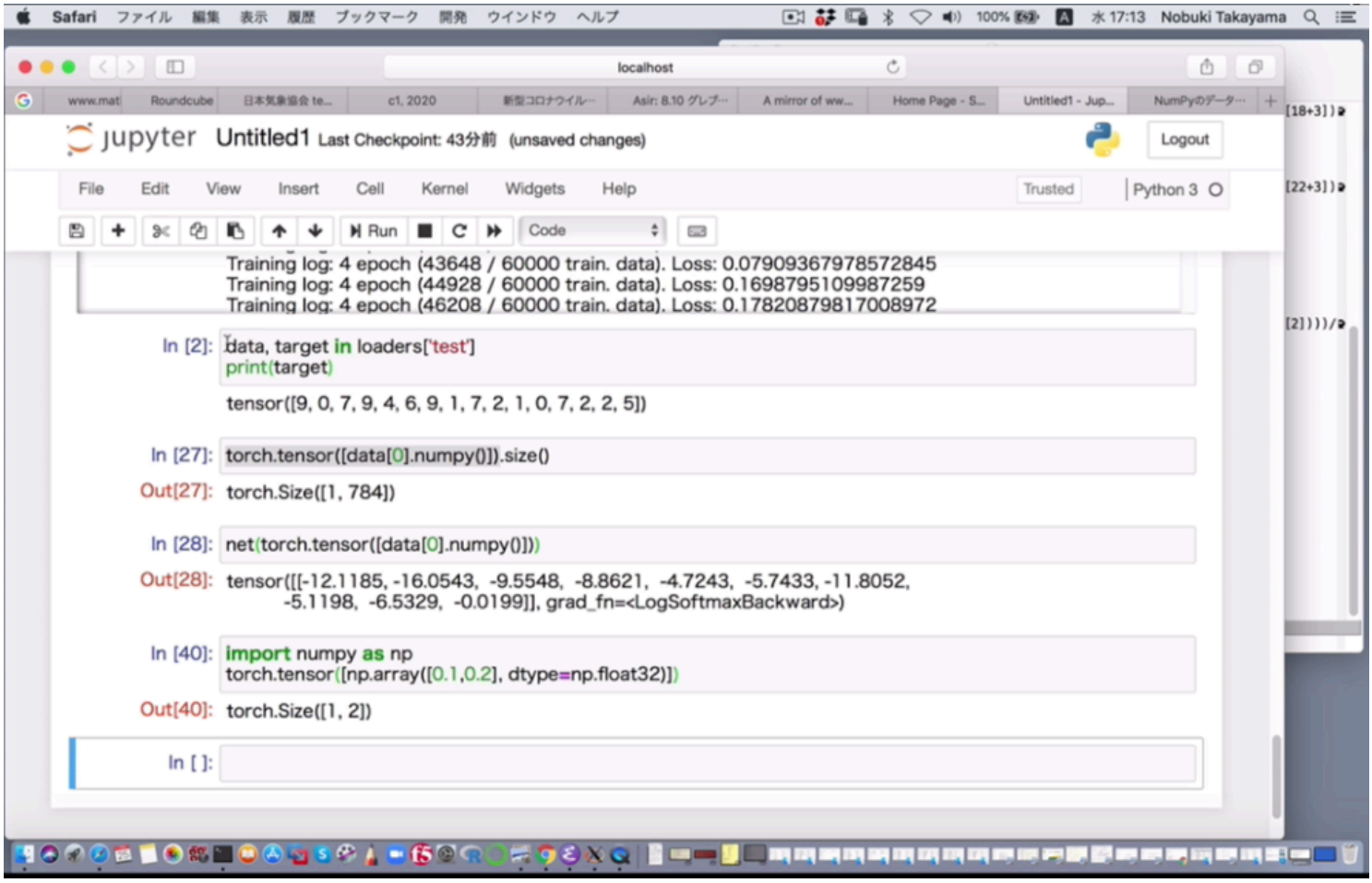
0800.png


BMP形式のデータを作る手頃なお絵かきソフトが windows 10 にあったのでそれを使う.
paint と下に入力
さっきまでは Mac の画面.
jupyter notebook は Windows 版もあり.

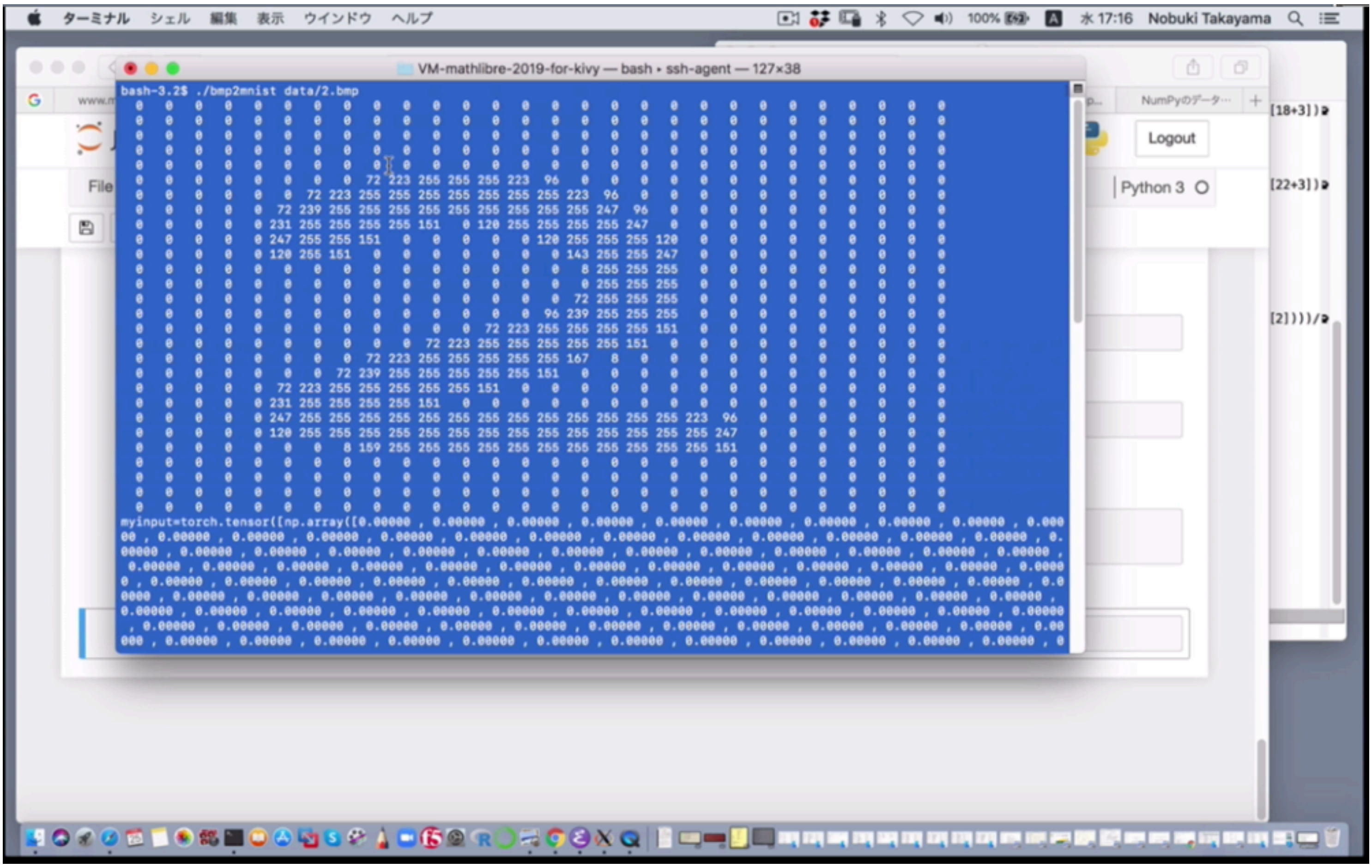
0820.png


ごみ箱
may-remove
z-Done
Wolfram Mathematica 12
PyTorch-MNIST-手書き文字の表示
VM
Math-notebooks
misc
asir-book-2017
ファイル
ホーム
表示
拡大
縮小
100%
ルーラー
グリッド線
ステータス バー
ズーム
表示または非表示
名前を付けて保存
PC › デスクトップ ›
デスクトップの検索
整理
新しいフォルダー
work
z-Done
アプリ
カメラアップロード
相続太郎丸
太郎丸
特殊関数セミナー
OneDrive
PC
Byobu
Math-notebooks
may-remove
misc
Todo
VM
z-Done
ファイル名(N): 2
ファイルの種類(T): PNG (*.png)
モノクロ ビットマップ (*.bmp;*.dib)
16 色ビットマップ (*.bmp;*.dib)
256 色ビットマップ (*.bmp;*.dib)
24 ビット ビットマップ (*.bmp;*.dib)
JPEG (*.jpg;*.jpeg;*.jpe;*.jfif)
GIF (*.gif)
TIFF (*.tif;*.tiff)
PNG (*.png)
HEIC (*.heic)
フォルダーの非表示
28 × 28px
800%
ここに入力して検索
15:56
2020/11/25

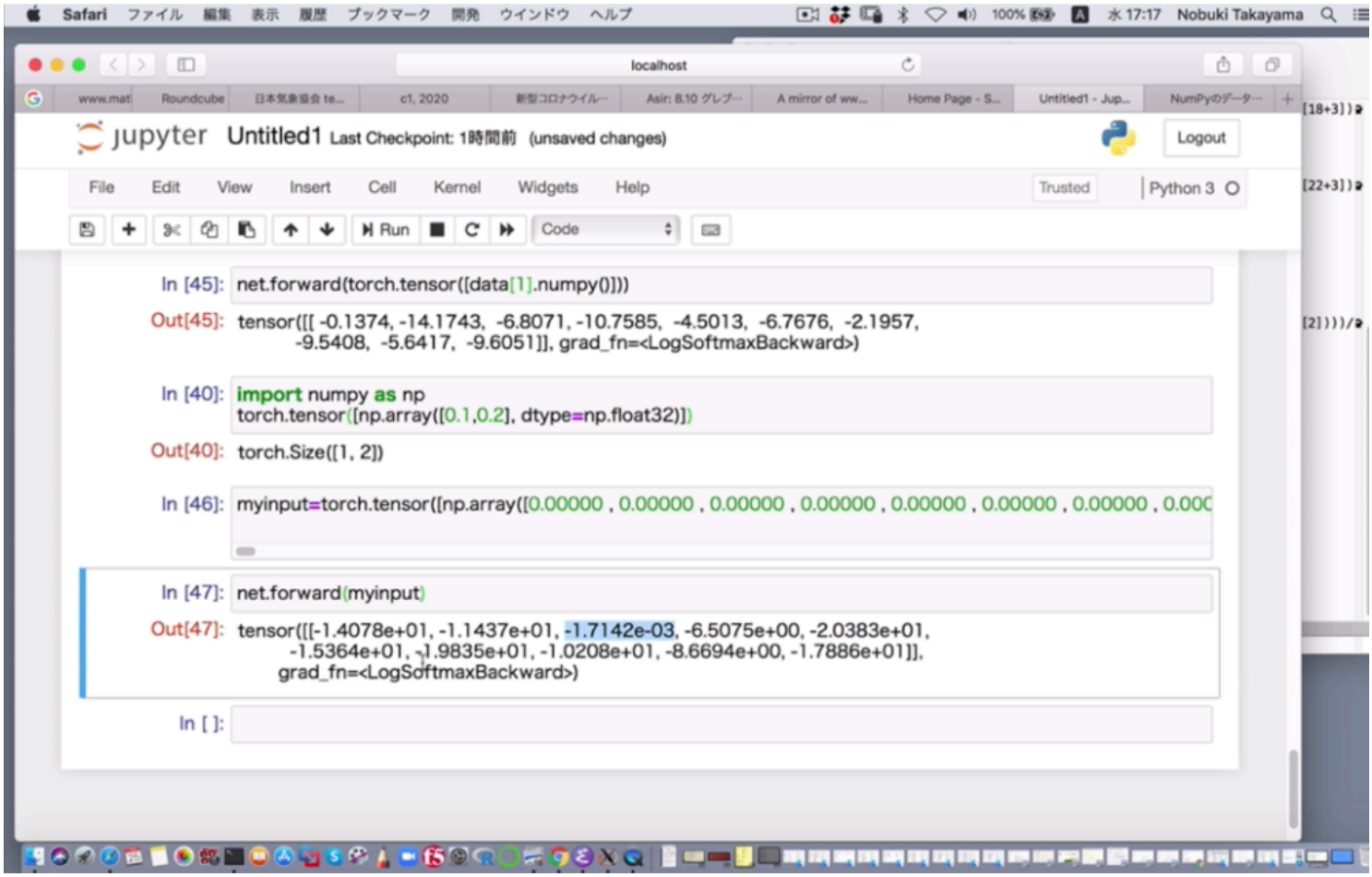
0830.png


Safari ファイル 編集 表示 履歴 ブックマーク 開発 ウインドウ ヘルプ
100% 水 17:13 Nobuki Takayama
localhost
jupyter Untitled1 Last Checkpoint: 43分前 (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted Python 3
Run Code
Training log: 4 epoch (43648 / 60000 train. data). Loss: 0.07909367978572845
Training log: 4 epoch (44928 / 60000 train. data). Loss: 0.1698795109987259
Training log: 4 epoch (46208 / 60000 train. data). Loss: 0.17820879817008972
In [2]: data, target in loaders['test']
print(target)
tensor([9, 0, 7, 9, 4, 6, 9, 1, 7, 2, 1, 0, 7, 2, 2, 5])
In [27]: torch.tensor([data[0].numpy()]).size()
Out[27]: torch.Size([1, 784])
In [28]: net(torch.tensor([data[0].numpy()]))
Out[28]: tensor([[-12.1185, -16.0543, -9.5548, -8.8621, -4.7243, -5.7433, -11.8052,
-5.1198, -6.5329, -0.0199]], grad_fn=<LogSoftmaxBackward>)
In [40]: import numpy as np
torch.tensor([np.array([0.1,0.2], dtype=np.float32)])
Out[40]: torch.Size([1, 2])
In [ ]:

0840.png


VM-mathlibre-2019-for-kivy — bash ▸ ssh-agent — 127×38
bash-3.2$ ./bmp2mnist data/2.bmp
myinput=torch.tensor([np.array([0.00000 , 0.00000 , 0.00000 , 0.00000 , 0.00000 , 0.00000 , 0.00000 , 0.00000 , 0.00000 , 0.000
Logout
Python 3
Nobuki Takayama

0850.png

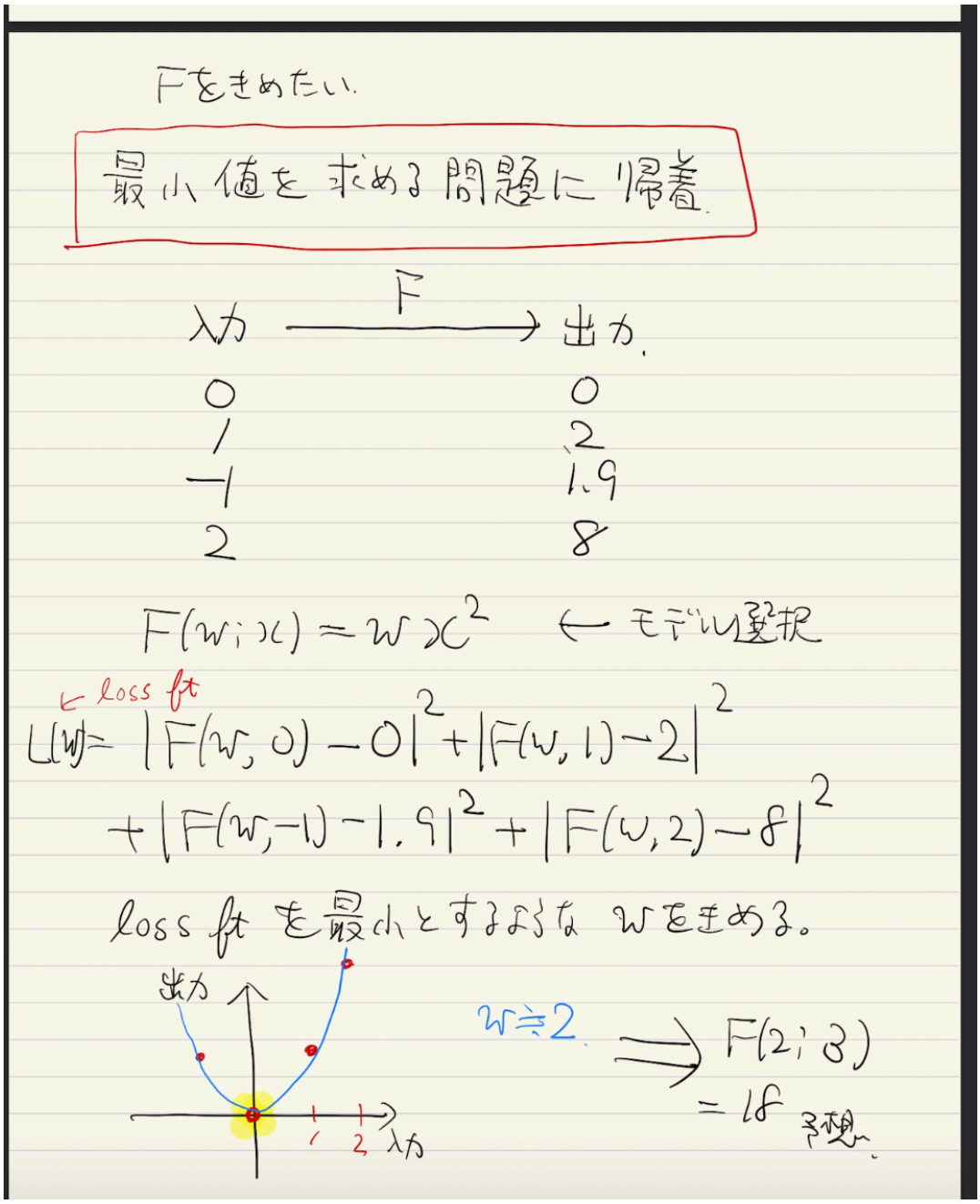


以下は補足

0860.png

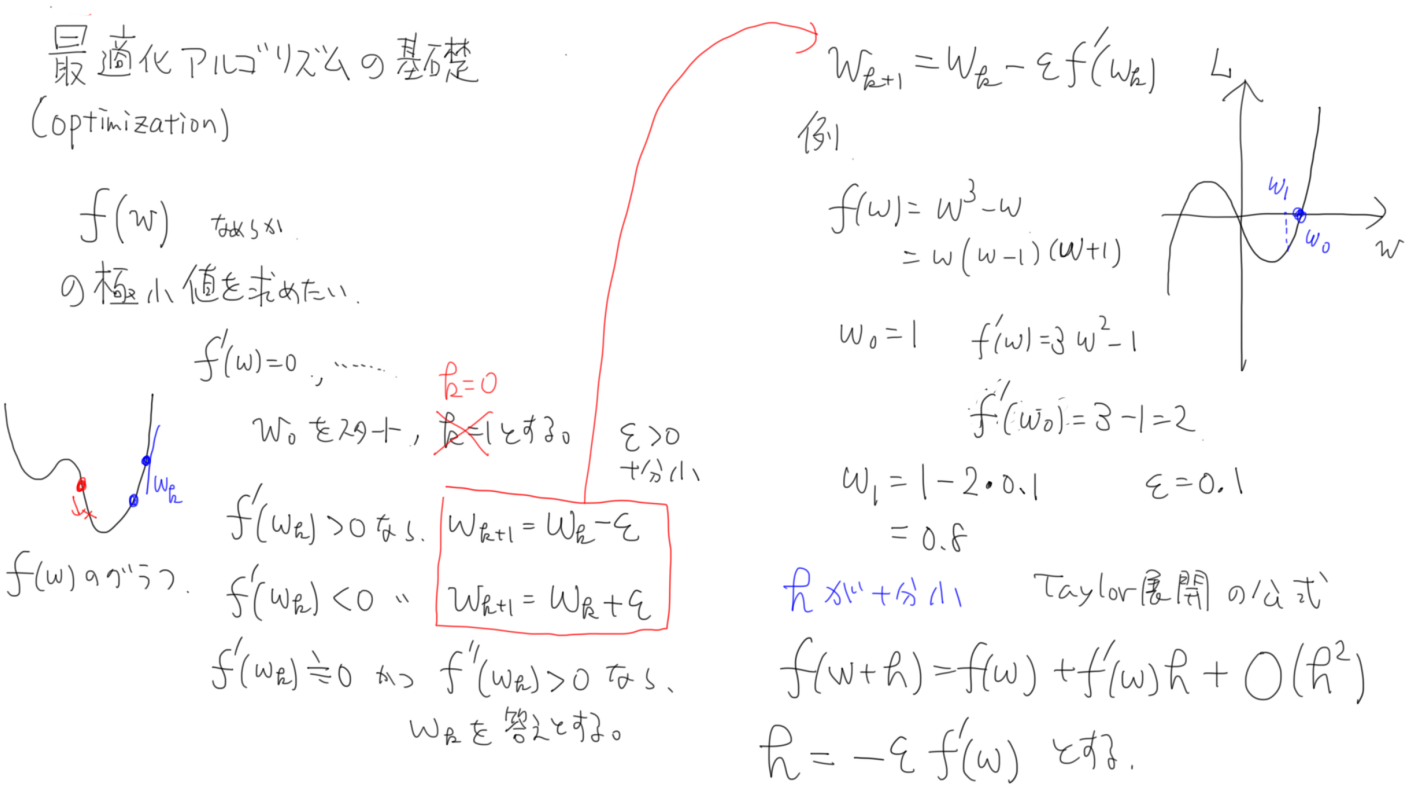
Movie in the mp4 format.

1000.png

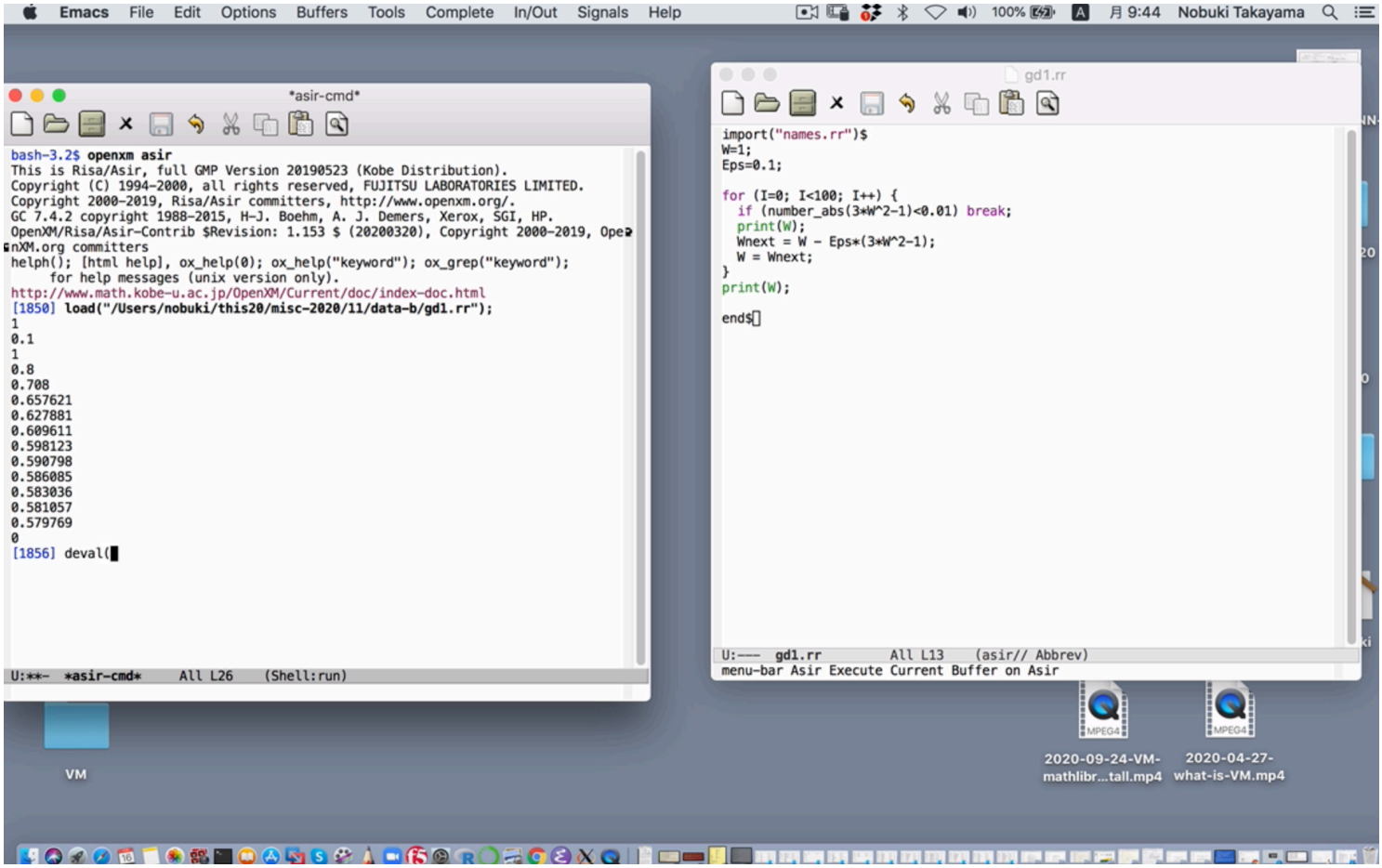


Movie in the mp4 format.
動画: 勾配降下法(gradient descent)の基礎 (youtube, 限定公開).

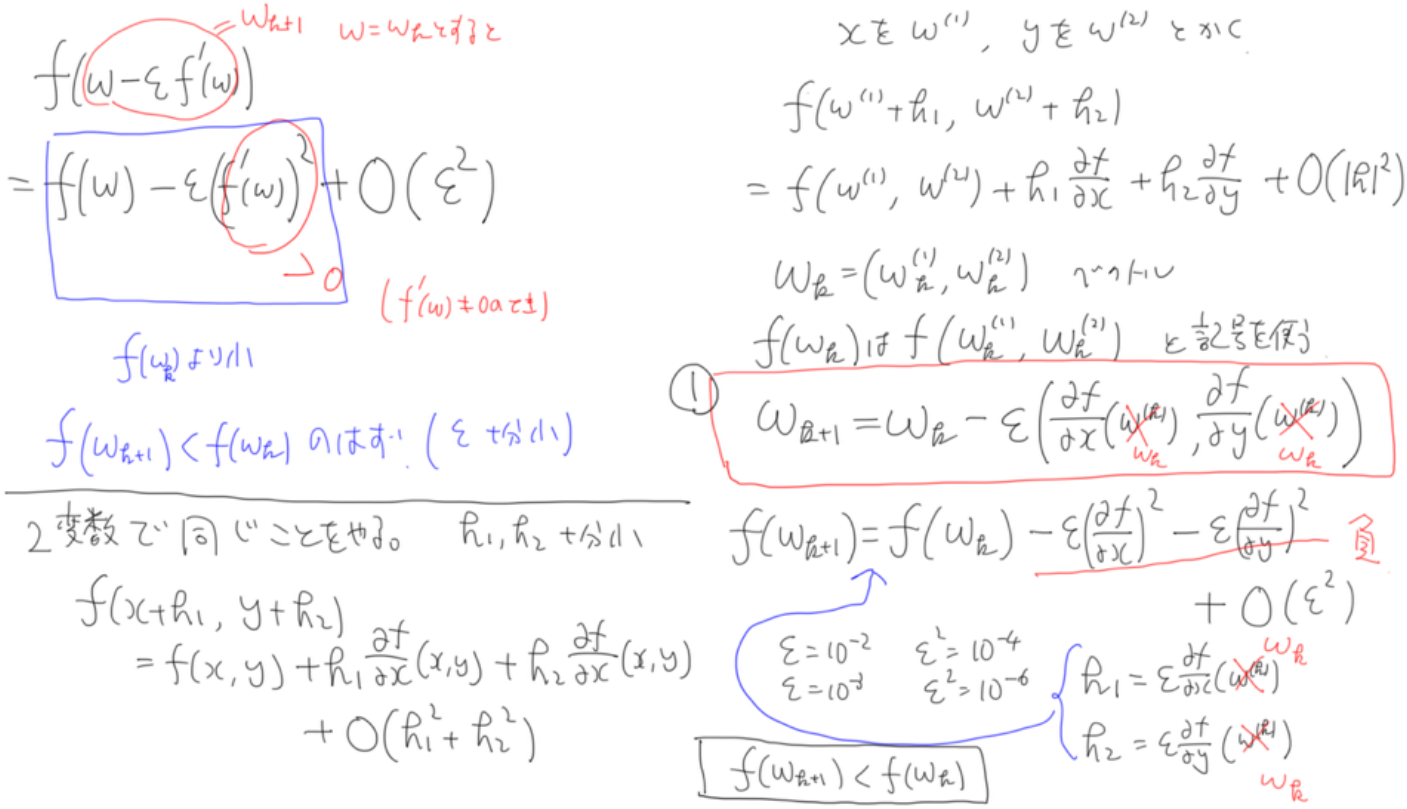
10000.png



10050.png


Emacs File Edit Options Buffers Tools Complete In/Out Signals Help
Nobuki Takayama
asir-cmd
bash-3.2$ openxm asir
This is Risa/Asir, full GMP Version 20190523 (Kobe Distribution).
Copyright (C) 1994-2000, all rights reserved, FUJITSU LABORATORIES LIMITED.
Copyright 2000-2019, Risa/Asir committers, http://www.openxm.org/.
GC 7.4.2 copyright 1988-2015, H-J. Boehm, A. J. Demers, Xerox, SGI, HP.
OpenXM/Risa/Asir-Contrib $Revision: 1.153 $ (20200320), Copyright 2000-2019, OpenXM.org committers
help(); [html help], ox_help(0); ox_help("keyword"); ox_grep("keyword");
for help messages (unix version only).
http://www.math.kobe-u.ac.jp/OpenXM/Current/doc/index-doc.html
[1850] load("/Users/nobuki/this20/misc-2020/11/data-b/gd1.rr");
1
0.1
1
0.8
0.708
0.657621
0.627881
0.609611
0.598123
0.590798
0.586085
0.583036
0.581057
0.579769
0
[1856] deval(
U:**- *asir-cmd* All L26 (Shell:run)
gd1.rr
import("names.rr")$
W=1;
Eps=0.1;
for (I=0; I<100; I++) {
if (number_abs(3*W^2-1)<0.01) break;
print(W);
Wnext = W - Eps*(3*W^2-1);
W = Wnext;
}
print(W);
ends$
U:--- gd1.rr All L13 (asir// Abbrev)
menu-bar Asir Execute Current Buffer on Asir
VM
2020-09-24-VM-mathlibr...tall.mp4
2020-04-27-what-is-VM.mp4

10100.png


$f(w-\varepsilon f'(w))$
$= f(w) - \varepsilon (f'(w))^2 + O(\varepsilon^2)$
$f(w_k)$ より小
$f(w_{k+1}) < f(w_k)$ のはず (ε十分小)
2変数で同じことをする。 h_1, h_2 十分小
$f(x+h_1, y+h_2) = f(x,y) + h_1 \frac{\partial f}{\partial x}(x,y) + h_2 \frac{\partial f}{\partial x}(x,y) + O(h_1^2 + h_2^2)$
$w=w_k$ とすると
w_{k+1}
$(f'(w) \neq 0$ とする)
$\to 0$
x を $w^{(1)}$, y を $w^{(2)}$ とおく
$f(w^{(1)}+h_1, w^{(2)}+h_2)$
$= f(w^{(1)}, w^{(2)}) + h_1 \frac{\partial f}{\partial x} + h_2 \frac{\partial f}{\partial y} + O(|h|^2)$
$w_k = (w_k^{(1)}, w_k^{(2)})$ ベクトル
$f(w_k)$ は $f(w_k^{(1)}, w_k^{(2)})$ と同じ意味
(1) $w_{k+1} = w_k - \varepsilon \left(\frac{\partial f}{\partial x}(w_k), \frac{\partial f}{\partial y}(w_k) \right)$
$f(w_{k+1}) = f(w_k) - \varepsilon \left(\frac{\partial f}{\partial x}\right)^2 - \varepsilon \left(\frac{\partial f}{\partial y}\right)^2 + O(\varepsilon^2)$
$\varepsilon = 10^{-2}$ $\varepsilon^2 = 10^{-4}$
$\varepsilon = 10^{-3}$ $\varepsilon^2 = 10^{-6}$
$h_1 = \varepsilon \frac{\partial f}{\partial x}(w_k)$
$h_2 = \varepsilon \frac{\partial f}{\partial y}(w_k)$
$f(w_{k+1}) < f(w_k)$

10200.png

①を用いて、$f(w)$の極小を求める方法を、
gradient descent とよぶ。

$\left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right)$ fのgradient

---

効率わるい ⇒

共役勾配法 $f(x,y)$が2次式なら、真の解に収束。
conjugate gradient method

$$H = \begin{pmatrix} \frac{\partial^2 f}{\partial x^2} & \frac{\partial^2 f}{\partial x \partial y} \\ \frac{\partial^2 f}{\partial y \partial x} & \frac{\partial^2 f}{\partial y^2} \end{pmatrix}$$

$f(x,y)$が2次式のとき　Hの固有値はすべて正，異なる（仮定）

$p_k \in \mathbb{R}^2$　　$p_i H p_j = 0 \quad (i \neq j)$　　Hのw_1での値

$w_k \in \mathbb{R}^2$

$$w_{k+1} = w_k + \alpha_k p_k \qquad k=1,2$$

$$\alpha_k = -\frac{r_k^T p_k}{p_k^T H p_k}$$

$$r_k = (\nabla \phi)(w_k) \qquad \nabla \phi = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right)$$

一般の場合

w_2を新w_1として、上をくりかえす

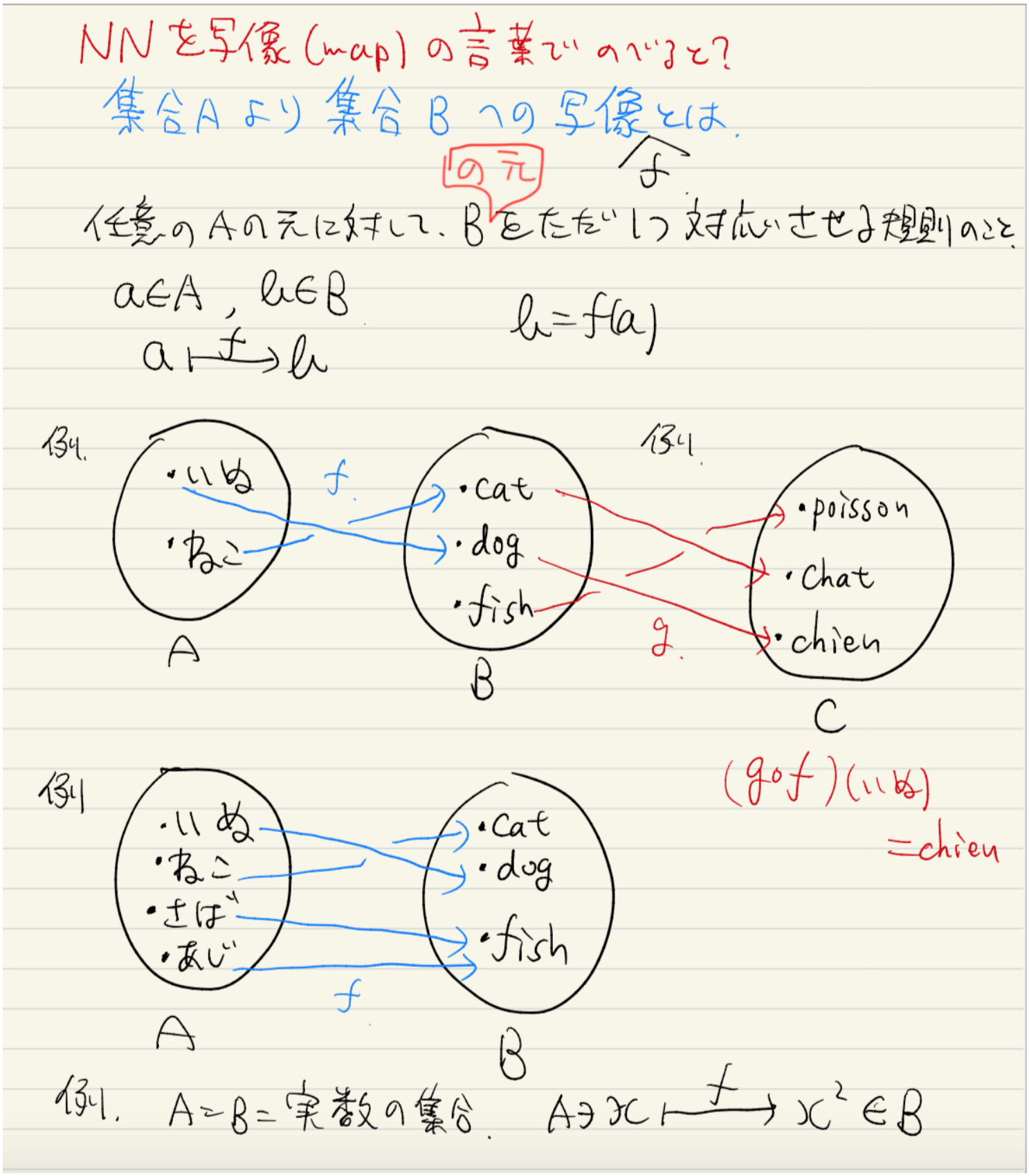
α_kはp_k方向に動かしてfが極小となる値に。

Hの計算を不要にする方法 いろいろあり

10300.png

Movie in the mp4 format.

0100.png



0200.png

例. $A=B=$平面上の点の集合.

$$A \ni (x,y) \xmapsto{f} (x+y, x-y) \in B$$

写像の合成

$f: A \to B$

$g: B \to C$ 対して.

$$A \ni a \longmapsto g(f(a)) \in C$$

↑ fとgの合成写像といい. $g \circ f$ と書く.

例. はた

パーセプトロンとは?

$$\mathbb{R}^n = \{(x_1, \cdots, x_n) \mid x_i \in \mathbb{R}\}$$

たとえば.

$$f_1: \mathbb{R}^{1000} \longrightarrow \mathbb{R}^{10}$$

線型写像

0300.png

$f_2 : \mathbb{R}^{1000} \longrightarrow \mathbb{R}^{1000}$ activation function

$f_3 : \mathbb{R}^{784} \longrightarrow \mathbb{R}^{1000}$ 線型写像

$f_1 \circ f_2 \circ f_3$ ← パーセプトロン

$f_i : \mathbb{R} \to \mathbb{R}$

$$\frac{d(f\circ g)}{dx} = \frac{df}{dx}(g(x))\frac{dg}{dx}$$

$x = p$

$q = g(p)$

$$(f\circ g)'(p) = f'(q)g'(p)$$

$$\frac{d}{dx}(f_1\circ f_2\circ f_3)(p) = ?$$

$q = f_3(p)$

$r = f_2(q)$

$s = f_1(r)$

$$\frac{d}{dx}\left(f_1\circ(f_2\circ f_3)\right)(p)$$

0400.png

$$= f_1'(r) \cdot (f_2 \circ f_3)'(p)$$

$$= f_1'(r) \cdot f_2'(q) \cdot f_3'(p)$$

backpropagation.

0500.png

微分積分で習う方法はダメ? そんなことはない. 数学ソフトウエアのアルゴリズムには微分積分で習う方法をさらに精緻にしたものがある. 解ける問題のサイズは小さいが正確な答えをだせる.

1. 一変数代数方程式, Schönhage algorithm. pari/gp の roots.
2. 多変数, Gröbner basis, たとえば
   `http://www.math.kobe-u.ac.jp/Asir/index-ja.html`
3. Numerical homotopy method, たとえば
   `http://homepages.math.uic.edu/~jan/download.html`

```
[1982] F;
5*x^10-5*x^8+x-5
[1983] G=diff(F,x);
50*x^9-40*x^7+1
[1984] pari(roots,G);
[ -0.9150100387698543982058464793395548142 1
  0.6608736761041111872598746977391170880 4
  0.8623159802759930519653427612509520687 2
  虚部のある解は省略.]
```