第３回ワークショップ
～コロナ禍における数理・データサイエンス・AI教育の取組：オンライン講義や教材活用など～

東京大学における取組

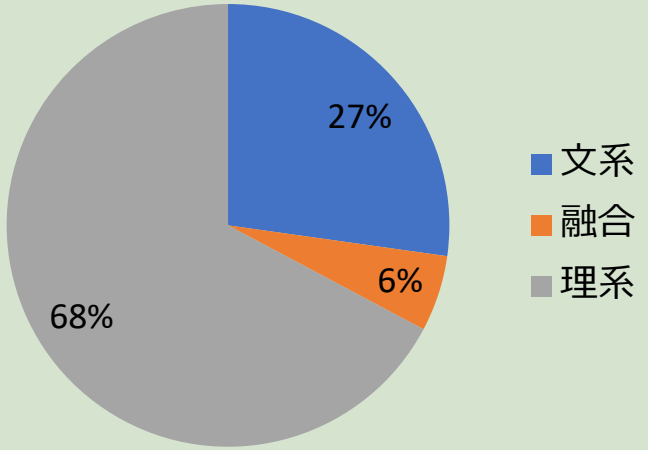
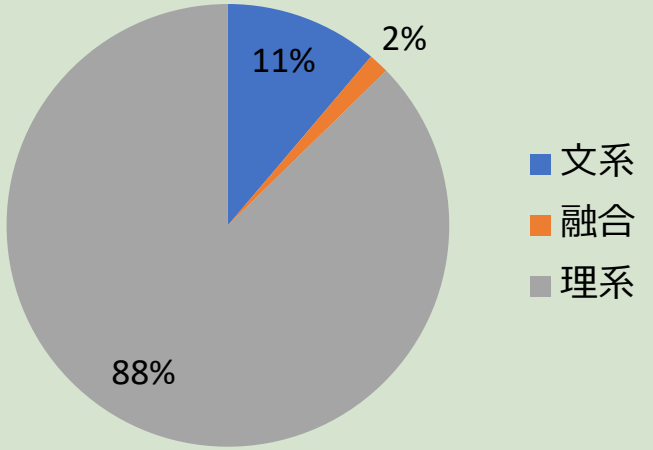
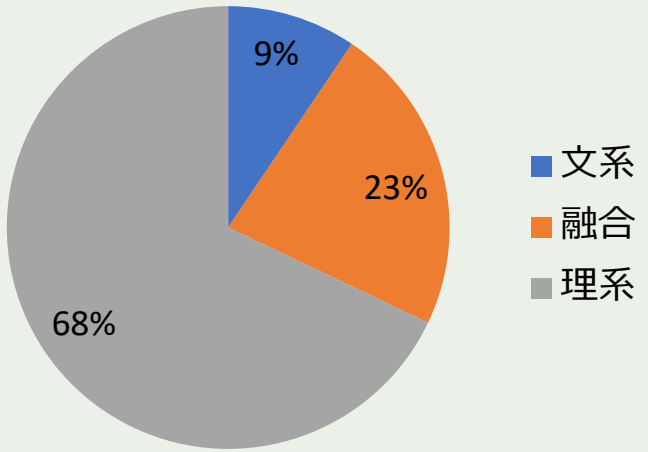
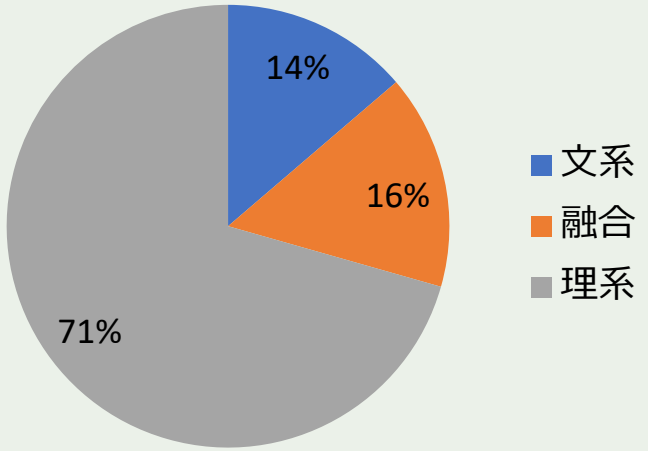
「Pythonプログラミング入門」
「メディアプログラミング入門」

東京大学大学院 情報理工学系研究科
数理・情報教育研究センター 准教授 山肩洋子

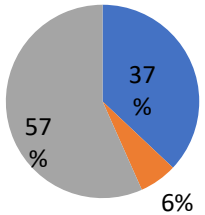
担当授業

	Pythonプログラミング入門	メディアプログラミング入門
回数	全7回（初回はガイダンス）	
内容	Pythonプログラミング基礎全般	音、テキスト、画像処理、Webスクレイピング等
担当者数	教員6名＋TA3~5名	教員1名＋TA2名
受講者数	S1期：約300名、SS(夏季集中):約35名 、A1:約280名	S2期：約185名
教材	対話的プログラミング環境 Jupyter notebook 実行環境はGoogle Colaboratoryを推奨	
授業形式	自宅で教材による予習を前提とした**反転授業** • 30分：課題の解説と総評 • 75分：Zoomブレイクアウトルームにて個別対応 （手を挙げた学生を一度に一人ずつ入室させる）	自宅で教材による予習＋**スライドで講義** • TAは採点のみ参加 • プログラミングのサポートは学習管理システム ITC-LMS の掲示板で対応
課題	毎回、予習課題・本課題を課す • 要求仕様に従い関数を定義させる内容 • Colaboratory上で解答し、そのリンクを提出 →学生は**自動採点システム**により**自分で評価**できる • 次回の授業にて模範解答や誤答例を提示	毎回、課題を課す • 教材に載っているコードを組み合わせれば解ける内容 →採点はスクリプトを作って半自動化 • 次回の授業にて模範解答を提示
授業HP	https://sites.google.com/view/ut-python	https://sites.google.com/view/ut-python-media/
一般公開 	教材（Jupyter notebook形式） https://utokyo-ipp.github.io/index.html	教材（Jupyter notebook形式）＋講義映像 http://www.mi.u-tokyo.ac.jp/teaching_material.html

受講生の文系・理系の割合（2020年S期）

	Pythonプログラミング入門	メディアプログラミング入門
学部生 **(3年生以上)**	180名 	134名 
院生	106名 	51名 

参考：東大全学部生の文・理割合



Pythonプログラミング入門の課題例

ex1ja.ipynb ☆

ファイル 編集 表示 挿入 ランタイム ツール ヘルプ 最終保存: 14:26

+ コード + テキスト

第1回本課題

Ex1-1. 三角形の分類

三角形の三辺の長さとなるべき正数 a, b, c を受け取り、

- 三辺が三角形を構成しないならば 0
- 三辺が構成する三角形が鋭角三角形ならば1
- 三辺が構成する三角形が直角三角形ならば2
- 三辺が構成する三角形が鈍角三角形ならば3

を返す関数 classify_triangle(a, b, c) を定義してください。

```
############################################################
##  <[ ex1-1-classify_triangle ]> 解答セル (Answer cell)
##  このコメントの書き変えを禁ず (Never edit this comment)
############################################################

def classify_triangle(a, b, c):
    ...
```

ここにコードを書く

提出前に以下のテストセルを実行し、 True のみが出力されることを確認してください。

```
[ ]  print(classify_triangle(3,6,5) == 3)
     print(classify_triangle(3,4,5) == 2)
     print(classify_triangle(4,3,3) == 1)
     print(classify_triangle(1,1,3) == 0)
     print(classify_triangle(1,2,1) == 0)
     print(classify_triangle(3,2,1) == 0)
```

コードが正しければ、このセルを実行するとすべてTrueとなる

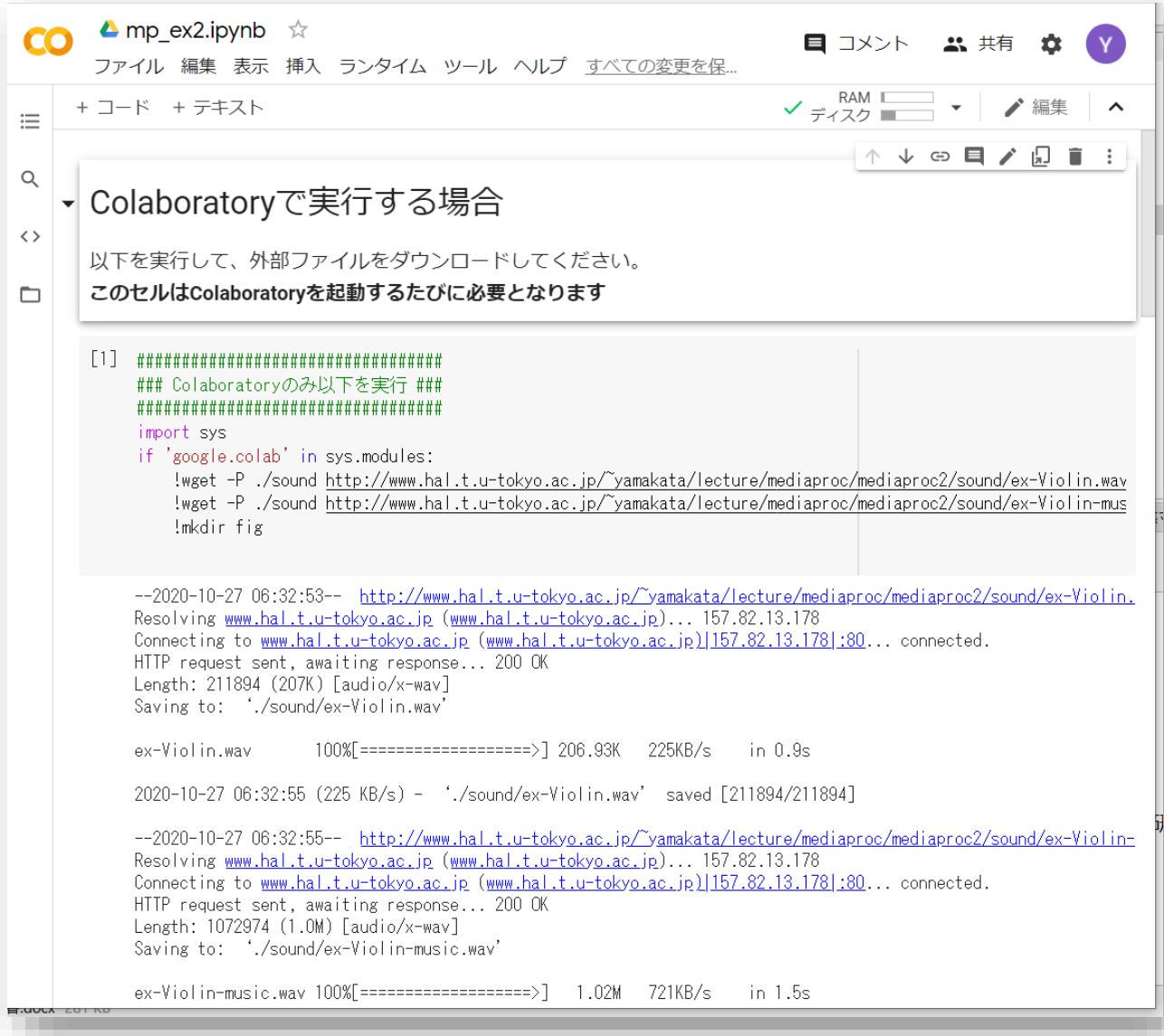
模範解答例

```
def classify_triangle(a, b, c):
    # 最長辺を求める
    if a < c < b or c < a <= b:
        a, b, c = a, c, b
    elif  b < c < a or c < b <= a:
        a, b, c = b, c, a
    # 任意のソート手続き, max(), 未習だがsorted,sortでも可
    if a>b:
        a, b = b, a # 未習なので代入文繰り返しでもよい
    if b>c:
        b, c = c, b # 未習なので代入文繰り返しでもよい

    # 三角形の成立条件と余弦定理による分岐
    if a + b > c:
        det = a**2 + b**2 - c**2
        if det > 0:
            return 1
        elif det == 0:
            return 2
        elif det < 0:
            return 3
    else:
        return 0
```

- 回を進むごとに考えさせる問題へ
- 入出力の一致で自動採点
  （競技プログラミングのマナーを踏襲）

外部入力ファイル（音、テキスト、画像データ）はサーバからダウンロード



最初のセルを実行するよう指示

音やテキスト、画像など
外部ファイルをサーバからダウンロード

メディアプログラミング入門の課題例

mp_ex2.ipynb ☆

ファイル 編集 表示 挿入 ランタイム ツール ヘルプ すべての変更を保...

コメント 共有 Y

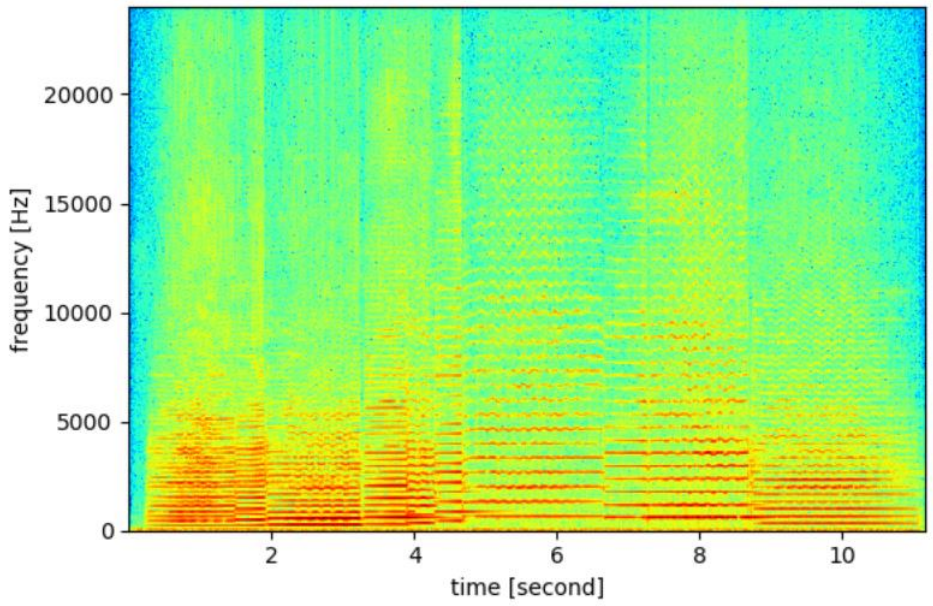
+ コード + テキスト RAM ディスク 編集

課題２：スペクトログラムの描画

ヴァイオリンの音楽 sound/Violin-music.wav のスペクトログラムを描画し、 ex2_2.png という名前の画像として保存してください。
スペクトログラム計算時のパラメータは以下の通りです（これは教材と同じです）。

- 窓幅は1024
- 移動窓における重なり（オーバーラップ）は64フレーム
- 窓関数はハミング窓

次のようなグラフになったらOKです。



模範解答例

```
%matplotlib inline

import wave
from pylab import *
import numpy as np

wavfile = wave.open("sound/ex-Violin-music.wav" , "rb" ) # オーディオファイルを開く

# wavfileのデータを最後まで一気に読み込む（データサイズが大きいファイルを読み込むときは、読み込むサイズを指定しないとPCがフリーズします）。
x = wavfile.readframes(wavfile.getnframes())
x = np.frombuffer(x, dtype= "int16") # バイナリデータをint型に変換
sampling_rate = wavfile.getframerate()
print(sampling_rate)
wavfile.close()

# 短く切り出すときの幅（切り出すフレームの数）
N = 1024

# FFTで用いるハミング窓
hammingWindow = np.hamming(N)

fig = plt.figure(figsize=(6, 4)) # figure(図を配置する画面)のサイズを指定

# スペクトログラムを描画
# cmap
pxx, freqs, bins, im = specgram(x,# 元の波形
                                NFFT=N, # 切り出す幅
                                Fs=sampling_rate, # サンプリングレート
                                noverlap=64, # 0から順番に切り出していくとき、重複して切り出すならここで重複させるフレーム数を指定
                                window=hammingWindow, # 窓関数の指定
                                cmap=cm.jet) # カラーマップ：値の大きさによって色を塗り分ける。
                                             # 様々なカラーマップが用意されている。
                                             # 参照：https://matplotlib.org/examples/color/colormaps_reference.html

xlabel("time [second]")
ylabel("frequency [Hz]")
fig.tight_layout() # 図がはみ出さないようにレイアウト
plt.savefig('fig/ex2_ans2.png') # 図を画像として保存
```

```
48000
```

- 教材から適切な部分をコピペすれば8割できる
- コロナ禍で「簡単すぎる」「考える課題がやりたい」というコメントがむしろ増えた
- ぜひ「考える課題」を出したいが、
  採点がとてつもなく大変になることが悩み
  （今はスクリプトを作って半自動採点）

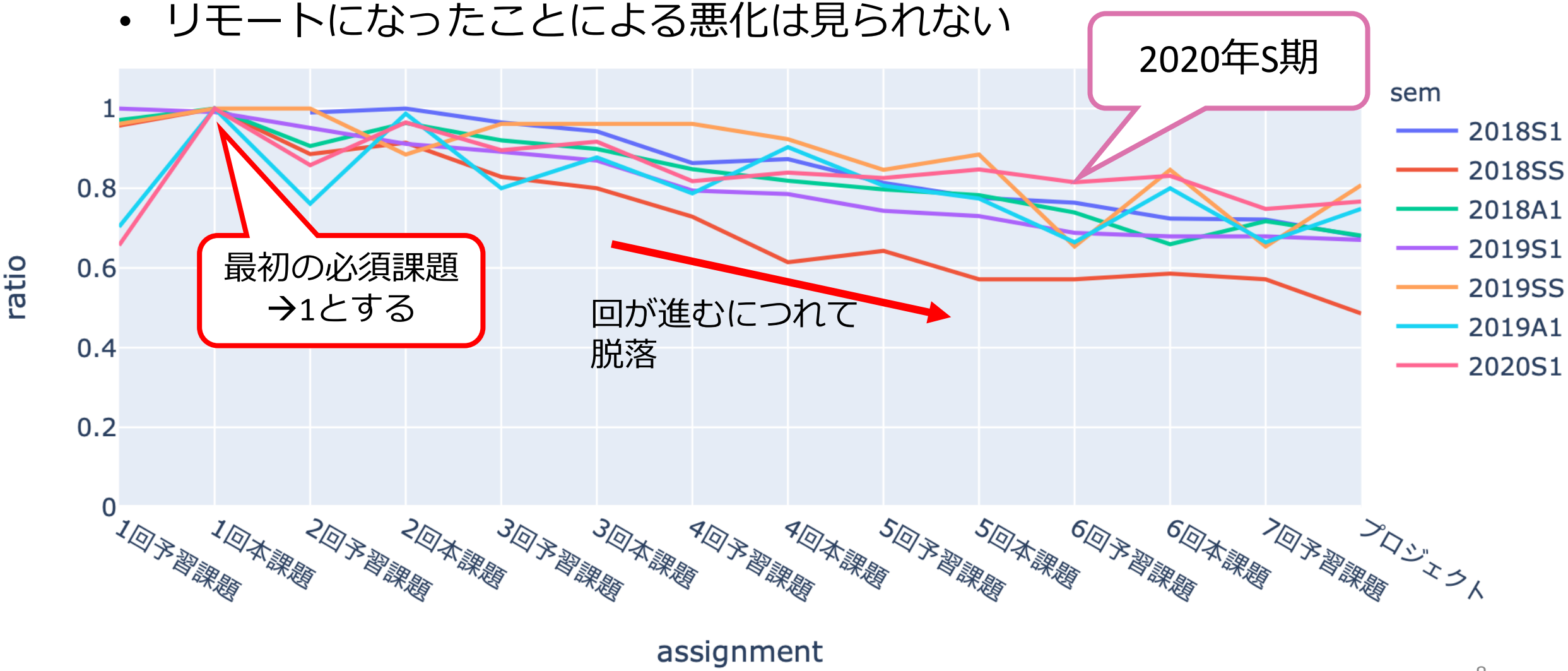
学生の反応：例年よりも若干いい

- オンラインのデメリットよりも、録画ビデオを提供したことで、都合のいい時間に受講できるようになったことのメリットが大
- プログラミング環境構築のトラブルはColaboratoryで回避
  - 前年度まではノートPC内に環境構築→オンラインでは不可能
  - 今年度からはGoogle Colaboratoryを推奨 → ほぼトラブルなし

    特に「メディアプログラミング入門」
    - パッケージのインストール時に多発していたトラブルがなくなった
    - 深層学習による画像認識はノートPCのCPUでは実行できない
      → ColaboratoryのGPUを使用することで可能に
- 発展課題のクオリティが向上
  - いずれも最終回で自由に考えさせる課題を出している（オプション）
  - 例年に比べユニークな解答が多発（外出自粛で時間があった？）

脱落曲線
（課題提出率＝課題提出者数/受講者数）

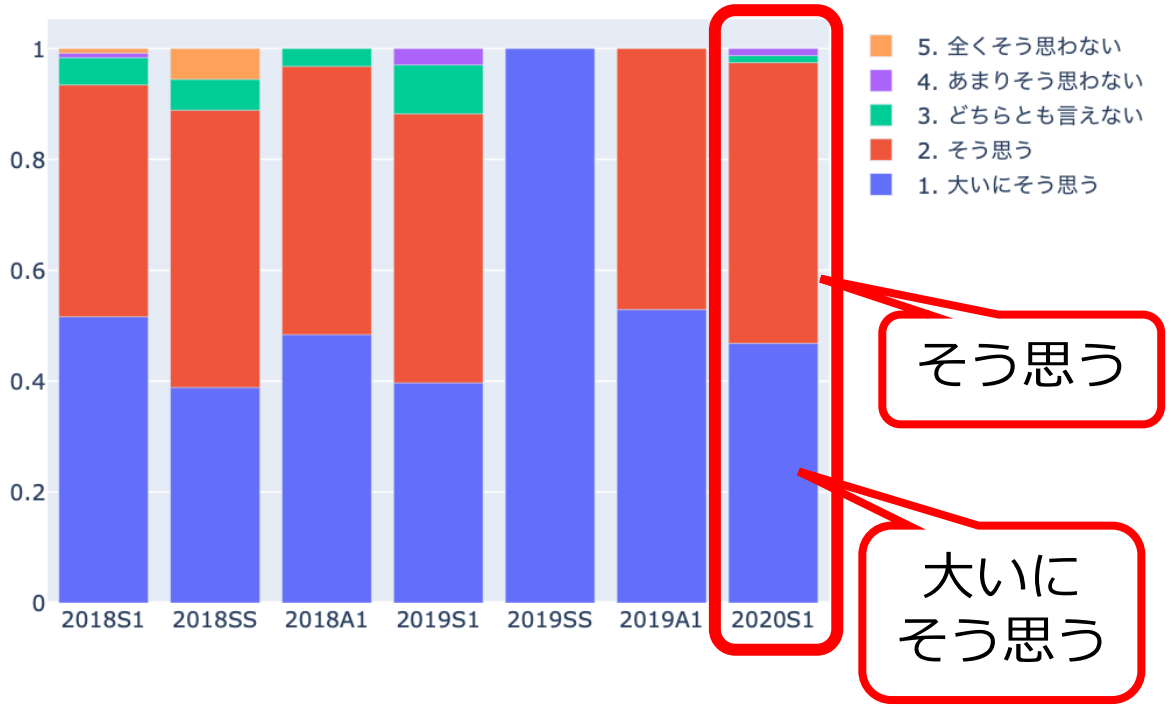
- 「Pythonプログラミング入門」
- リモートになったことによる悪化は見られない



受講生アンケートの評価の推移(2018S期～)

- 「Pythonプログラミング入門」
- リモートになったことによる悪化は見られない

総合的に高く評価できるか



教員の熱意が感じられたか