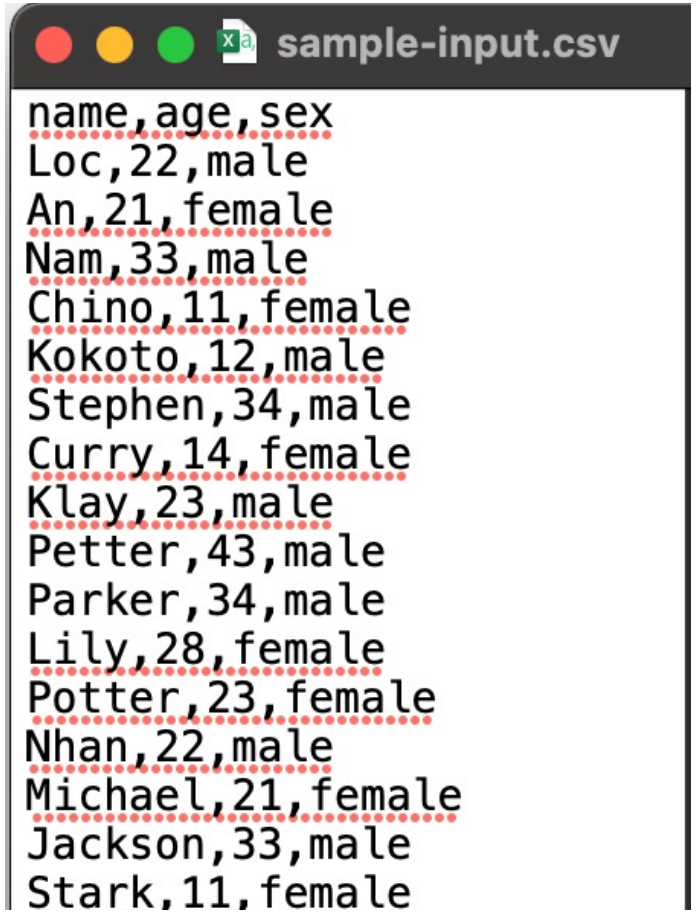
Python with CSV

Index

- CSV format
- CSV library
- Pandas library
- Pandas vs CSV performance
- Conclusion

What is .csv format

What is CSV format?



	name	age	sex
1	Loc	22	male
2	An	21	female
3	Nam	33	male
4	Chino	11	female
5	Kokoto	12	male
6	Stephen	34	male
7	Curry	14	female
8	Klay	23	male
9	Petter	43	male
10	Parker	34	male
11	Lily	28	female
12	Potter	23	female
13	Nhan	22	male
14	Michael	21	female

- CSV stands for "comma separated values"
- Used to store the tabular data

CSV module in Python

CSV module in Python

- A build-in module in Python
- Provided 2 approaches to read/write .csv file:
  - Default approach
  - Dictionary approach

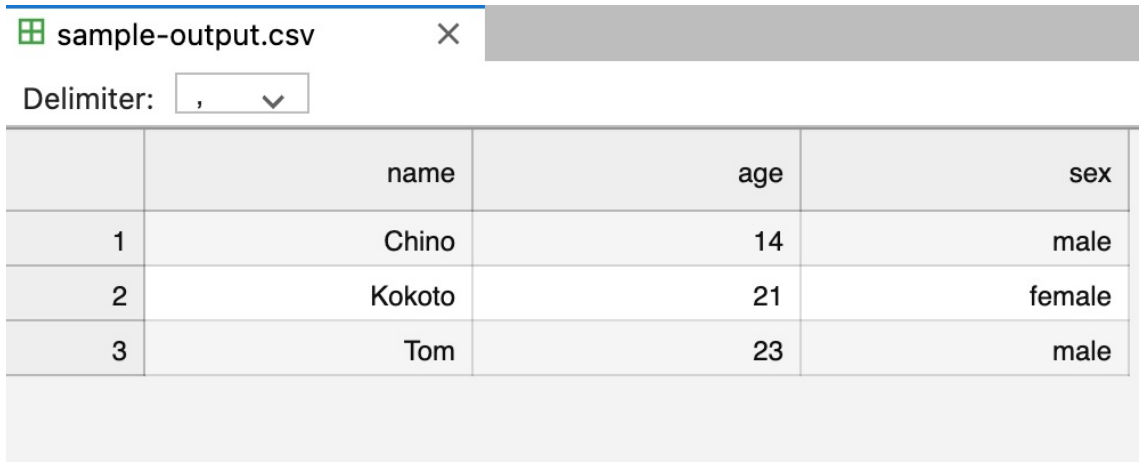
csv.reader

```python
with open(file="sample-input.csv", mode="r") as csv_file:
    csv_reader = csv.reader(csv_file)
    header = next(csv_reader)

    print("Header:", header)
    print("Data:")
    for line in csv_reader:
        print(line)

```

```
Header: ['name', 'age', 'sex']
Data:
['Loc', '22', 'male']
['An', '21', 'female']
['Nam', '33', 'male']
['Chino', '11', 'female']
['Kokoto', '12', 'male']
['Stephen', '34', 'male']
['Curry', '14', 'female']
```

csv.writer

```
data = [["name", "age", "sex"],
        ["Chino", 14, "male"],
        ["Kokoto", 21, "female"],
        ["Tom", 23, "male"]]

with open(file="sample-output.csv", mode="w") as csv_file:
    csv_writer = csv.writer(csv_file)
    csv_writer.writerows(data)
```



	name	age	sex
1	Chino	14	male
2	Kokoto	21	female
3	Tom	23	male

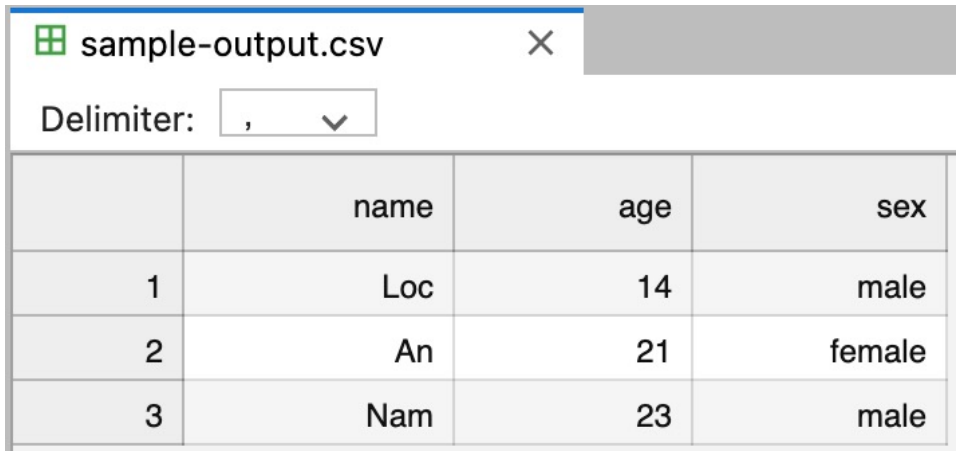
csv.DictReader

```python
with open(file="sample-input.csv", mode="r") as csv_file:
    csv_reader = csv.DictReader(csv_file)

    for line in csv_reader:
        print(line)
```

```
{'name': 'Loc', 'age': '22', 'sex': 'male'}
{'name': 'An', 'age': '21', 'sex': 'female'}
{'name': 'Nam', 'age': '33', 'sex': 'male'}
{'name': 'Chino', 'age': '11', 'sex': 'female'}
{'name': 'Kokoto', 'age': '12', 'sex': 'male'}
{'name': 'Stephen', 'age': '34', 'sex': 'male'}
{'name': 'Curry', 'age': '14', 'sex': 'female'}
```

csv.DictWriter

```python
data = [
    {'name': "Loc", 'age': 14, 'sex': "male"},
    {'name': "An", 'age': 21, 'sex': "female"},
    {'name': "Nam", 'age': 23, 'sex': "male"}
]

fieldnames = ["name", "age", "sex"]
with open(file="sample-output.csv", mode="w") as csv_file:
    csv_writer = csv.DictWriter(csv_file, fieldnames=fieldnames)
    csv_writer.writeheader()
    csv_writer.writerows(data)
```



	name	age	sex
1	Loc	14	male
2	An	21	female
3	Nam	23	male

Pandas module in Python

Pandas module library

- Pandas is not a build-in module
- By default, pandas use DataFrame to manipulate csv file

pandas.read_csv

```python
import pandas as pd

data = pd.read_csv("sample-input.csv")
print(type(data))
print(data)
```

```
<class 'pandas.core.frame.DataFrame'>
       name  age     sex
0       Loc   22    male
1        An   21  female
2       Nam   33    male
3     Chino   11  female
4    Kokoto   12    male
5   Stephen   34    male
6     Curry   14  female
7      Klay   23    male
8    Petter   43    male
9    Parker   34    male
10     Lily   28  female
```

pandas.DataFrame.to_csv

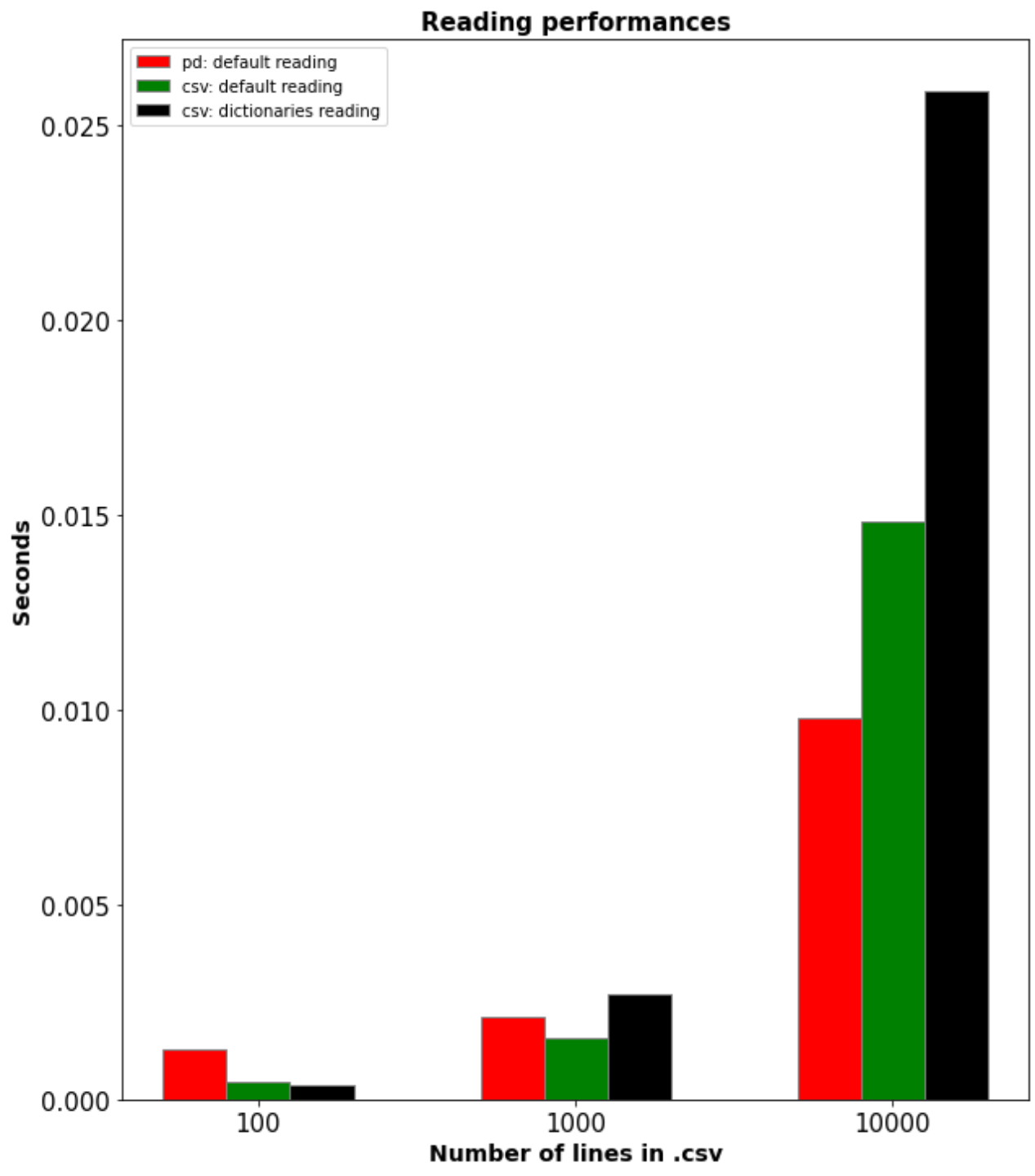
```python
import pandas as pd

data = pd.DataFrame(
    {
        'name': ['Loc', 'An', 'Nam'],
        'age': [22, 19, 25],
        'sex': ['male', 'male', 'female']
    }
)
data.to_csv("sample-output.csv", index=False)

```

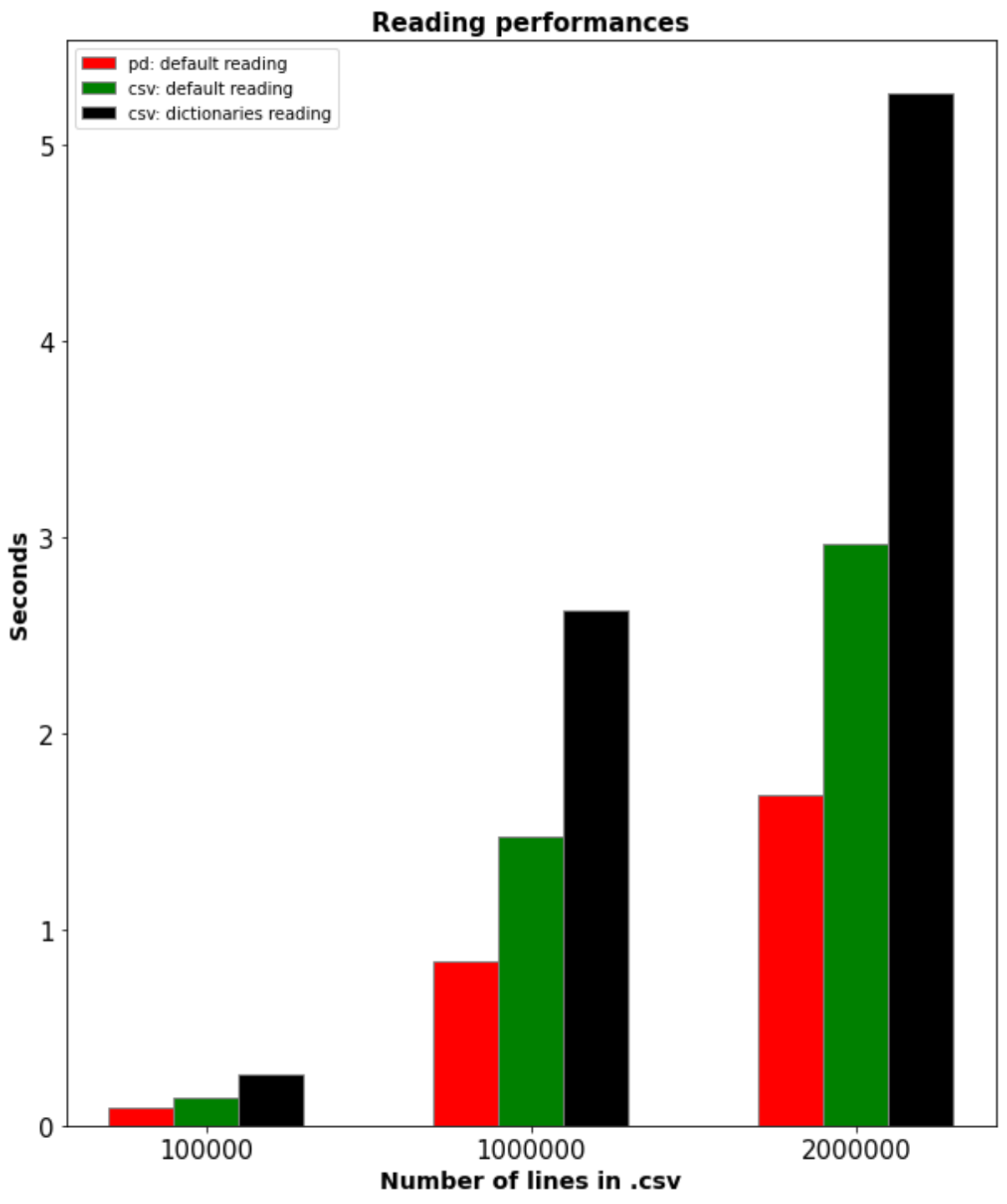
sample-output.csv

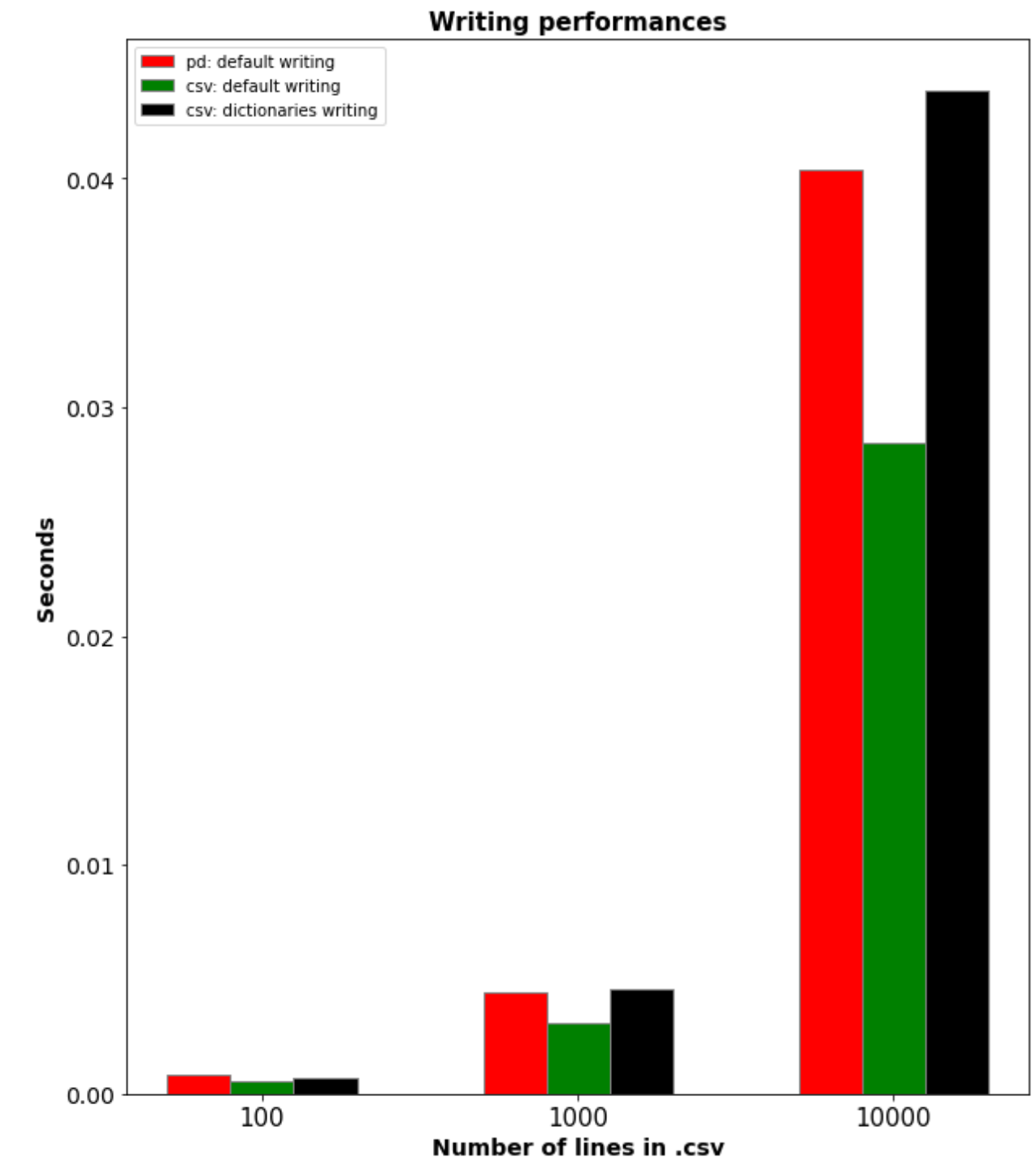
Delimiter: ,

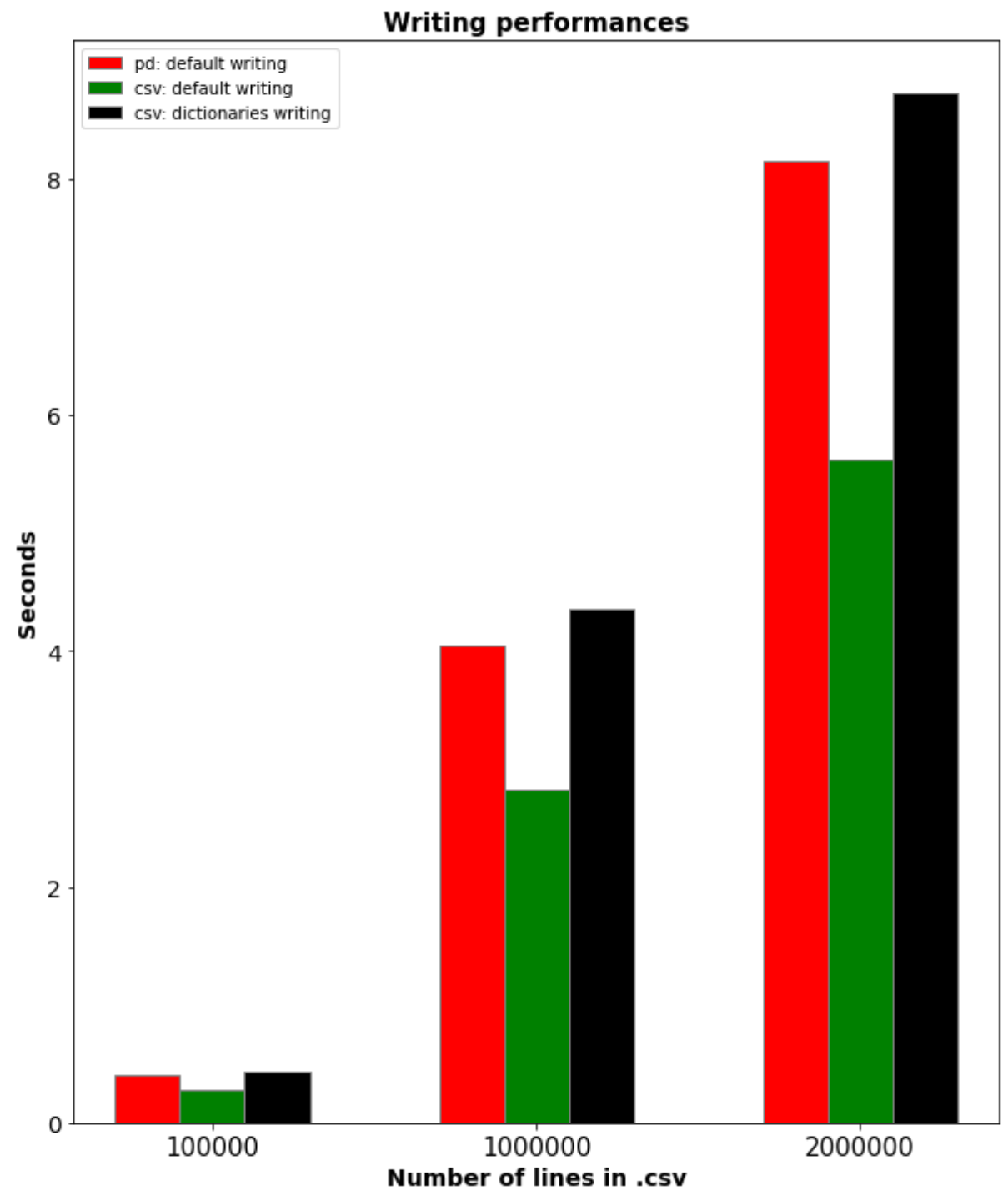
	name	age	sex
1	Loc	22	male
2	An	19	male
3	Nam	25	female

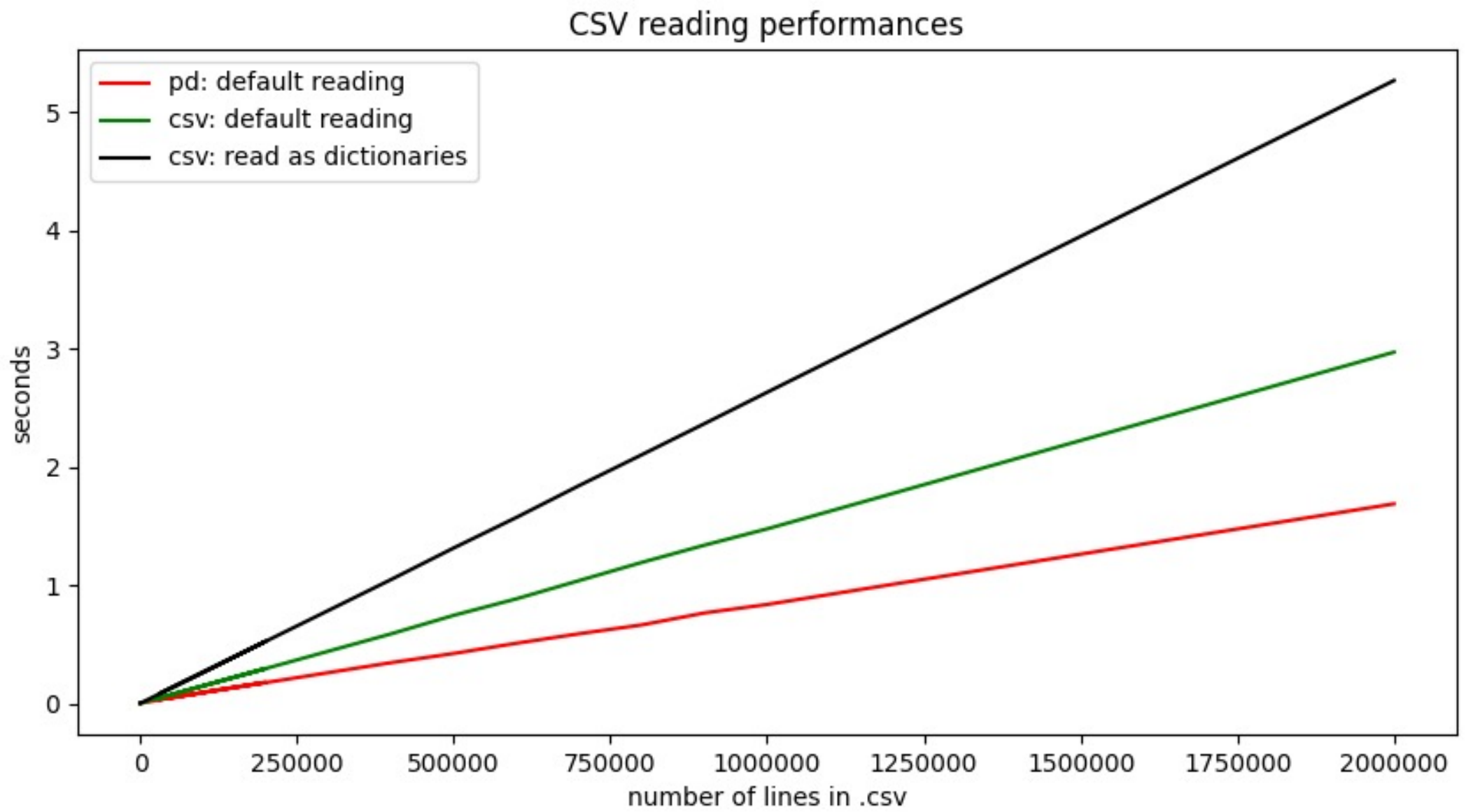
Pandas vs CSV performance

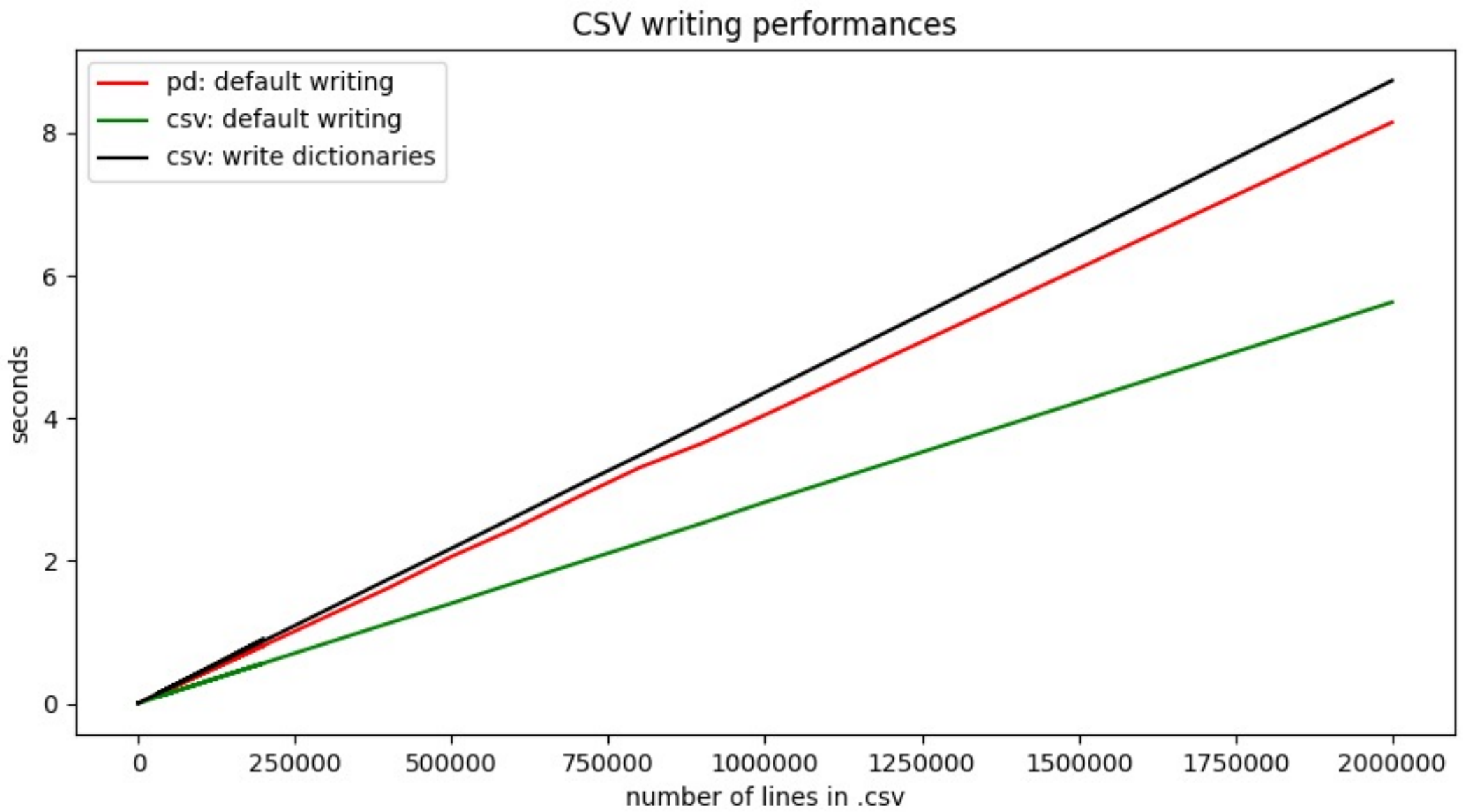

Reading performances
pd: default reading
csv: default reading
csv: dictionaries reading
Seconds
0.000
0.005
0.010
0.015
0.020
0.025
100
1000
10000
Number of lines in .csv


Reading performances
pd: default reading
csv: default reading
csv: dictionaries reading
Seconds
0
1
2
3
4
5
100000
1000000
2000000
Number of lines in .csv


Writing performances
pd: default writing
csv: default writing
csv: dictionaries writing
0.04
0.03
0.02
0.01
0.00
Seconds
100
1000
10000
Number of lines in .csv


Writing performances
pd: default writing
csv: default writing
csv: dictionaries writing
8
6
4
2
0
Seconds
100000
1000000
2000000
Number of lines in .csv


CSV reading performances
pd: default reading
csv: default reading
csv: read as dictionaries
seconds
0
1
2
3
4
5
0
250000
500000
750000
1000000
1250000
1500000
1750000
2000000
number of lines in .csv


CSV writing performances
pd: default writing
csv: default writing
csv: write dictionaries
seconds
0
2
4
6
8
0
250000
500000
750000
1000000
1250000
1500000
1750000
2000000
number of lines in .csv

Reading performances

- pandas.read_csv is slower than csv.reader when the csv file is less than 5000 lines
- For csv file with more than 5000 lines, pandas.read_csv is significantly faster than csv.reader, and therefore faster than csv.DictReader

Writing performances

- pandas.DataFrame.to_csv is slower than csv.writer
- pandas.DataFrame.to_csv is faster than csv.DictWriter

pandas vs csv module

pandas	csv module
Is not a built-in module → increases project dependencies	Is a built-in module of Python
Appropriate for small to big data	Appropriate for small size csv
Can be applied for .csv, excel, json, html, sql file	Only applied for .csv format