mmCEsim Simulation Example

Wuqiong Zhao

This is a basic millimeter wave channel estimation simulation example with mmCEsim. The involved algorithms are 'OMP' and 'Oracle LS'. There are 4 jobs in total, with SNR and pilot overhead as variables and NMSE as metric. The PFD report is auto generated via 'simreport.cls' and a corresponding plain text report is also available.

Contents

1 System Settings

The simulation adoptes the geometric channel model for millimeter wave (mmWave).

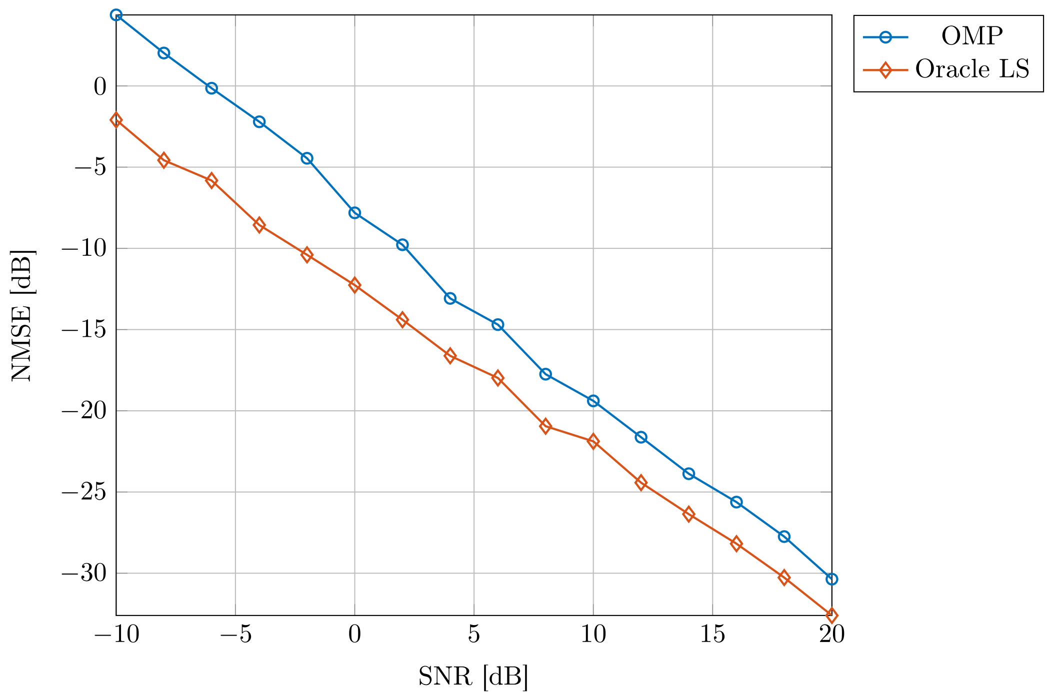
Name	Antenna Number	Beam Number	Grid Number
Transmitter	8×1	2×1	8×1
Receiver	16×1	4×1	16×1

- Channel Sparsity: 6;
- Off Grid Effect: false;
- Bandwidth: Narrowband.

2 Simulation Results

2.1 NMSE v.s. SNR (Pilot: 32)

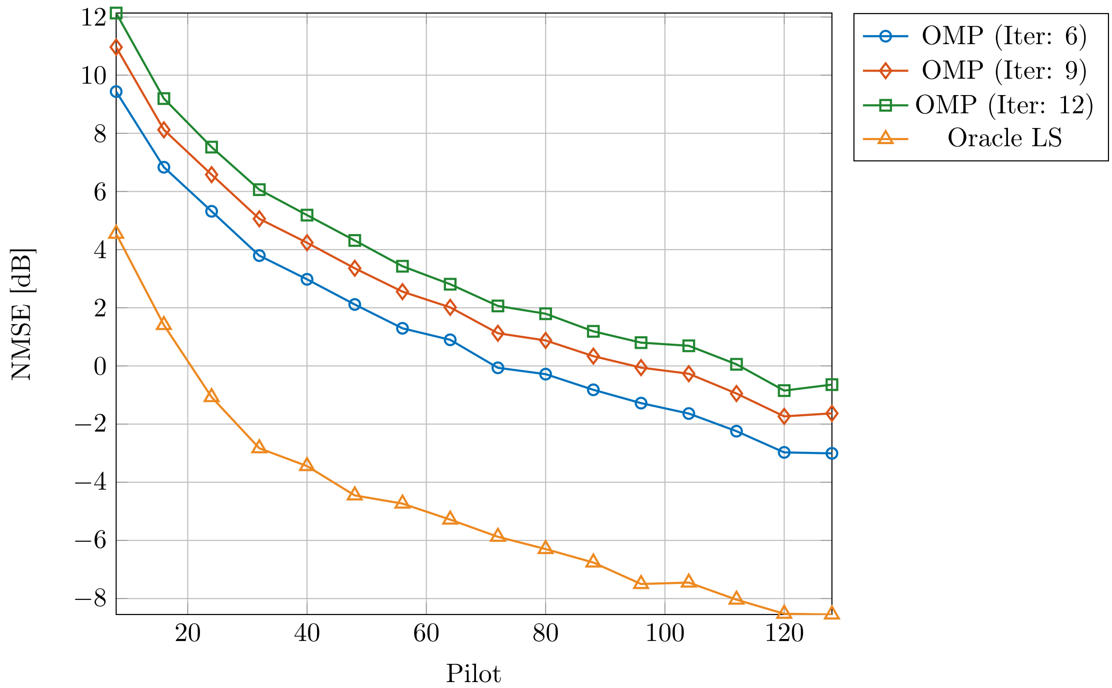
SNR [dB]	OMP	Oracle LS
−10	4.37	−2.09
−8	2.03	−4.57
−6	−0.14	−5.82
−4	−2.21	−8.56
−2	−4.45	−10.40
0	−7.81	−12.26
2	−9.78	−14.39
4	−13.08	−16.62
6	−14.69	−17.98
8	−17.75	−20.95
10	−19.39	−21.88
12	−21.63	−24.43
14	−23.87	−26.36
16	−25.62	−28.18
18	−27.74	−30.27
20	−30.37	−32.60



Simulated with 100 Monte Carlo tests.

2.2 NMSE v.s. Pilot (-10 dB)

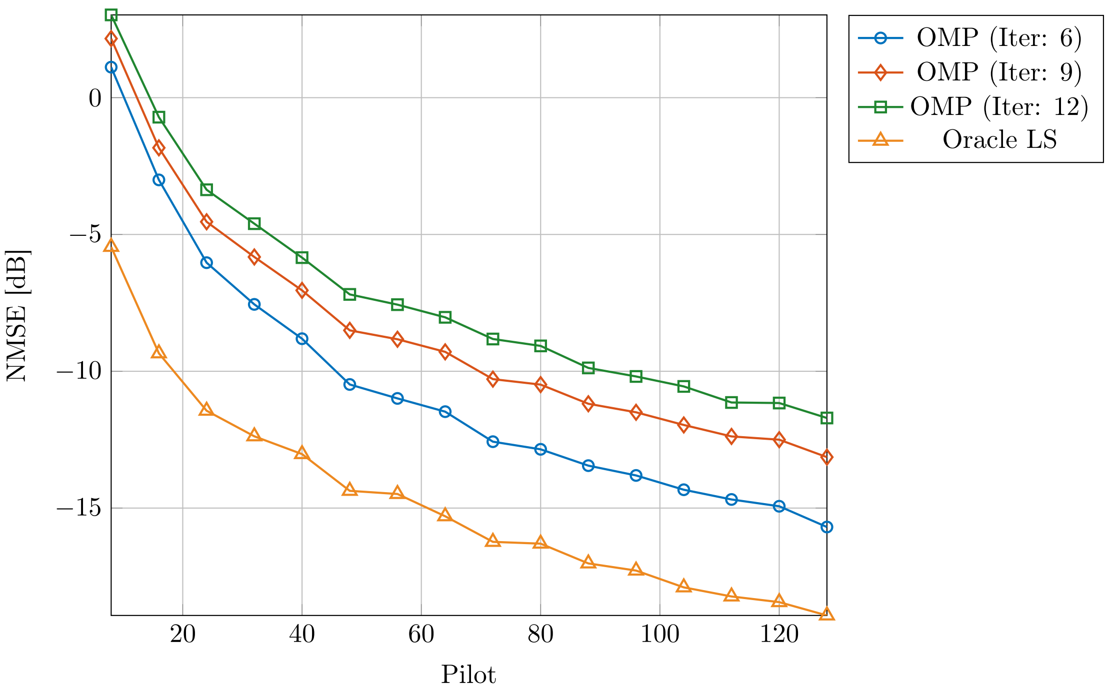
Pilot	OMP (Iter: 6)	OMP (Iter: 9)	OMP (Iter: 12)	Oracle LS
8	9.44	10.96	12.14	4.55
16	6.83	8.12	9.20	1.41
24	5.32	6.58	7.53	−1.07
32	3.79	5.06	6.06	−2.83
40	2.98	4.24	5.19	−3.44
48	2.11	3.36	4.32	−4.45
56	1.29	2.56	3.43	−4.73
64	0.90	2.01	2.81	−5.28
72	−0.06	1.12	2.06	−5.88
80	−0.28	0.88	1.80	−6.30
88	−0.82	0.34	1.19	−6.76
96	−1.28	−0.06	0.80	−7.50
104	−1.64	−0.27	0.69	−7.45
112	−2.24	−0.95	0.06	−8.04
120	−2.97	−1.73	−0.85	−8.52
128	−3.01	−1.63	−0.64	−8.55



Simulated with 200 Monte Carlo tests.

2.3 NMSE v.s. Pilot (0 dB)

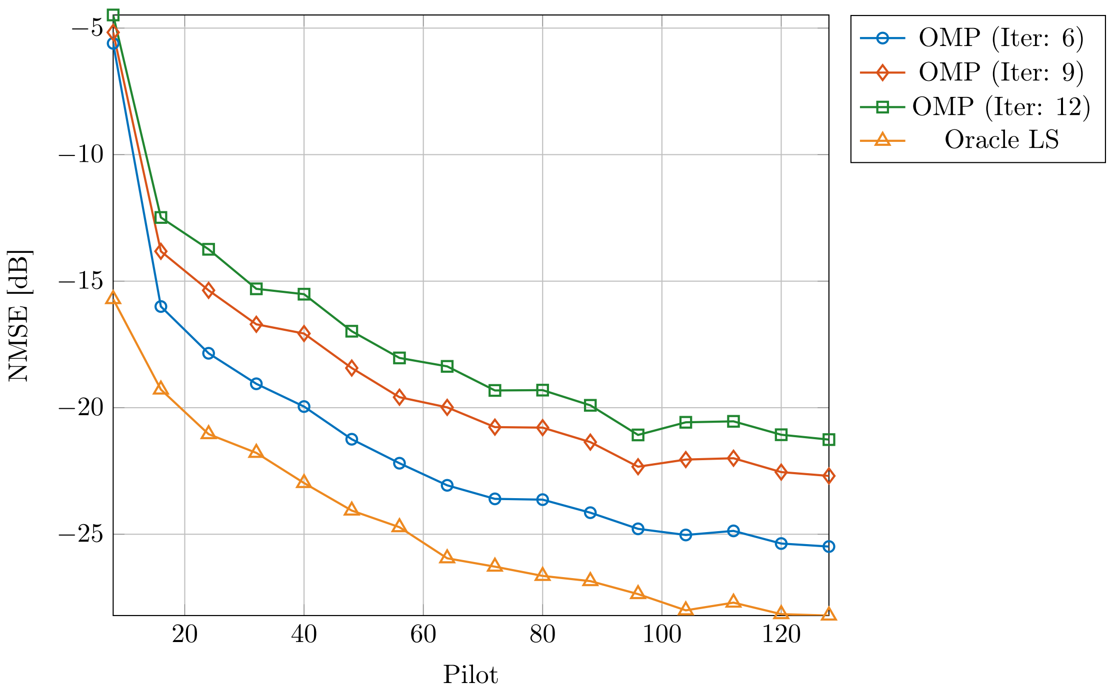
Pilot	OMP (Iter: 6)	OMP (Iter: 9)	OMP (Iter: 12)	Oracle LS
8	1.12	2.16	3.03	−5.45
16	−3.01	−1.83	−0.71	−9.34
24	−6.03	−4.54	−3.37	−11.44
32	−7.56	−5.82	−4.61	−12.38
40	−8.81	−7.04	−5.85	−13.03
48	−10.48	−8.50	−7.19	−14.37
56	−10.99	−8.83	−7.56	−14.49
64	−11.48	−9.29	−8.03	−15.30
72	−12.57	−10.29	−8.82	−16.24
80	−12.85	−10.49	−9.07	−16.30
88	−13.45	−11.19	−9.88	−17.02
96	−13.81	−11.50	−10.19	−17.28
104	−14.33	−11.96	−10.55	−17.90
112	−14.69	−12.38	−11.14	−18.23
120	−14.93	−12.50	−11.16	−18.44
128	−15.69	−13.14	−11.71	−18.92



Simulated with 200 Monte Carlo tests.

2.4 NMSE v.s. Pilot (10 dB)

Pilot	OMP (Iter: 6)	OMP (Iter: 9)	OMP (Iter: 12)	Oracle LS
8	−5.60	−5.17	−4.48	−15.71
16	−16.00	−13.82	−12.48	−19.28
24	−17.85	−15.36	−13.74	−21.05
32	−19.06	−16.70	−15.30	−21.78
40	−19.96	−17.07	−15.51	−22.98
48	−21.25	−18.43	−16.98	−24.06
56	−22.19	−19.58	−18.03	−24.72
64	−23.07	−19.99	−18.37	−25.95
72	−23.60	−20.77	−19.32	−26.28
80	−23.63	−20.79	−19.31	−26.65
88	−24.15	−21.36	−19.91	−26.85
96	−24.79	−22.33	−21.08	−27.37
104	−25.03	−22.05	−20.58	−28.01
112	−24.87	−22.00	−20.54	−27.70
120	−25.37	−22.54	−21.07	−28.16
128	−25.49	−22.70	−21.26	−28.21



Simulated with 200 Monte Carlo tests.

3 Simulation Configuration

3.1 Configuration File

Listing 1: `Example_Configuration.sim`

```
# Example_Configuration.sim
# mmCEsim Simulation Example
# Author: Wuqiong Zhao
# Date: 2022-09-20

version: 0.1.0 # the targeted mmCEsim version
meta: # document meta data
  title: mmCEsim Simulation Example
  description:
    This is a basic millimeter wave channel estimation simulation example with mmCEsim.
    The involved algorithms are `OMP' and `Oracle LS'.
    There are 4 jobs in total, with SNR and pilot overhead as variables and NMSE as metric.
    The PFD report is auto generated via `simreport.cls'
    and a corresponding plain text report is also available.
  author: Wuqiong Zhao
  email: contact@mmcesim.org
  website: https://mmcesim.org
  license: MIT
  date: "2022-09-18"
  comments: This is an uplink channel.
physics:
  frequency: narrow # assume narrow band
  off_grid: false # do not consider off-grid problem
nodes:
  - id: BS # this should be unique
    role: receiver
    num: 1 # this is the default value
    size: [16, 1] # UPA with size 8x4
    beam: [4, 1]
    grid: same # the same as physics size
    beamforming:
      variable: "W"
      scheme: random
  - id: UE # user
    role: transmitter
    num: 1 # a single-user model
    size: 8 # ULA with size 8
    beam: 2
    grid: 8
    beamforming:
      variable: "F"
      scheme: random
channels:
  - id: H
    from: BS
    to: UE # 'from -> to' specifies the channel direction
    sparsity: 6
    gains:
      mode: normal
      mean: 0
      variance: 1
sounding:
  variables:
    received: "y" # received signal vector
    noise: "noise" # received noise vector
```

```
    channel: "H_cascaded" # the cascaded channel (actually the same as 'H' for simple MIMO)
preamble: |
  COMMENT Here starts the preamble.
estimation: |
  VNt::m = NEW `DICTIONARY.T`
  VNr::m = NEW `DICTIONARY.R`
  lambda_hat = INIT `GRID.*`
  Q = INIT `MEASUREMENT` `GRID.*`
  i::u0 = LOOP 0 `PILOT`/`BEAM.T`
    F_t::m = NEW F_{:,:,i}
    W_t::m = NEW W_{:,:,i}
    Q_{i*`BEAM.*`:(i+1)*`BEAM.*`-1,:} = \kron(F_t^T, W_t^H) @ \kron(VNt^*, VNr) # the sensing matrix
  END
  none_zero::u1 = NEW \find(\abs(VNr^H@H_cascaded@VNt)>0.1)
  # PRINT \size(none_zero,0) '\n' # make sure the number of non-zero elements
  BRANCH
  lambda_hat = ESTIMATE Q y none_zero
  RECOVER VNr @ \reshape(lambda_hat, `GRID.R`, `GRID.T`) @ VNt^H
  MERGE
conclusion: |
  PRINT "">>\t"" `JOB_CNT` '\n'
simulation:
  backend: cpp # cpp (default) | matlab | octave | py
  metric: [NMSE] # used for compare
  jobs:
    - name: "NMSE v.s. SNR (Pilot: 32)"
      test_num: 100
      SNR: [-10:2:20]
      SNR_mode: dB # dB (default) | linear
      pilot: 32
      # pilot_mode: percent # num (default) | percent
      algorithms: # compare different languages
        - alg: OMP
          max_iter: 6
          label: OMP # used in report
          estimated_channel: H_hat_OMP # variable name for the estimated channel
        - alg: Oracle_LS
          label: Oracle LS
    - name: NMSE v.s. Pilot (-10 dB)
      test_num: 200
      SNR: -10
      pilot: [8:8:128]
      algorithms: # compare different languages
        - alg: OMP
          max_iter: 6
          label: "OMP (Iter: 6)"
        - alg: OMP
          max_iter: 9
          label: "OMP (Iter: 9)"
        - alg: OMP
          max_iter: 12
          label: "OMP (Iter: 12)"
        - alg: Oracle_LS
          label: Oracle LS # used in report
    - name: NMSE v.s. Pilot (0 dB)
      test_num: 200
      SNR: 0
      pilot: [8:8:128]
      algorithms: # compare different languages
        - alg: OMP
          max_iter: 6
          label: "OMP (Iter: 6)"
```

```
        - alg: OMP
          max_iter: 9
          label: "OMP (Iter: 9)"
        - alg: OMP
          max_iter: 12
          label: "OMP (Iter: 12)"
        - alg: Oracle_LS
          label: Oracle LS # used in report
    - name: NMSE v.s. Pilot (10 dB)
      test_num: 200
      SNR: 10
      pilot: [8:8:128]
      algorithms: # compare different languages
        - alg: OMP
          max_iter: 6
          label: "OMP (Iter: 6)"
        - alg: OMP
          max_iter: 9
          label: "OMP (Iter: 9)"
        - alg: OMP
          max_iter: 12
          label: "OMP (Iter: 12)"
        - alg: Oracle_LS
          label: Oracle LS # used in report
  report:
    name: mmCEsim_Example_Report
    format: [pdf, latex] # both compiled PDF and tex files
    plot: true # plot data
    table: false # do not print table
    latex:
      command: xelatex # command to compile the report
      UTF8: false # no need for UTF8 support with this setting
```

3.2 Algorithms

Listing 2: `OMP.alg`

```
#! Function: OMP
#! Description: Orthogonal matching pursuit compressed sensing.
#! Author: Wuqiong Zhao
#! Date: 2022-09-16
#! Version: 0.1.0

# Input:
#   - Q: Sensing matrix
#   - y: Received signal
#   - L: Sparsity
# Output:
#   - h: The estimated sparse signal
h::v = FUNCTION OMP Q::m y::v L::u0
  COMMENT Start of OMP algorithm!
  h = \zeros(\size(Q, 1)) # initialize as zeros
  Q_H::m = NEW Q^H # the conjugate transpose of Q
  r = NEW y # residual
  r_last::v = NEW r * 2 # the residual in last iteration
  support = INIT \length(y) dtype=u # over-length support array
  term = INIT $\size(Q_H, 0)$ dtype=f # float number array
  j::u0 = NEW 0
  a::v = INIT
  FOR "" $j != \length(y)$ $j = j + 1$
    term = \abs(Q_H @ r)
    index::u0 = NEW \index_max(term)
    IF \ismember(index, support)
```

```
      BREAK # end of the LOOP
    END
    support_{j} = index
    columns::m = NEW Q_{:, support_{0:j}}
    a = \pinv(columns) @ y
    r = y - columns @ a
    IF \sum(\abs(r - r_last)) / \sum(\abs(r_last)) < 0.0001 || j >= L
      j = j + 1
      BREAK # accurate enough to end iteration
    ELSE
      r_last = r
    END
  END
  # prepare for the final return
  h_{support_{0:j-1}} = a
END
```

Listing 3: `Oracle_LS.alg`

```
#! Function: Oracle_LS
#! Description: Oracle LS compressed sensing.
#! Author: Wuqiong Zhao
#! Date: 2022-09-18
#! Version: 0.1.0

h::v = FUNCTION Oracle_LS Q::m y::v indices::u1
  h = \zeros(\size(Q,1))
  h_{indices} = \pinv(Q_{:, indices}) @ y
END
```

4 mmCEsim Information

This report is auto generated by mmCEsim. The application **mmCEsim** is a powerful tool to simulate millimeter wave (mmWave) channel estimation (CE) for both experts and learners.

mmCEsim is *open source*! The software can be freely used and distributed under the MIT license.

- Official website: `https://mmcesim.org`
- Documentation: `https://mmcesim.org/doc`
- Tutorial: `https://mmcesim.org/tutorial`
- Web Application: `https://app.mmcesim.org`
- Blog: `https://blog.mmcesim.org`
- Publications: `https://pub.mmcesim.org`
- GitHub Organization: `https://github.com/mmcesim`
- Twitter: `https://twitter.com/mmcesim`
- VS Code Extension: `https://marketplace.visualstudio.com/items?itemName=mmcesim.mmcesim`