

Oneweb, Starlink and 5G

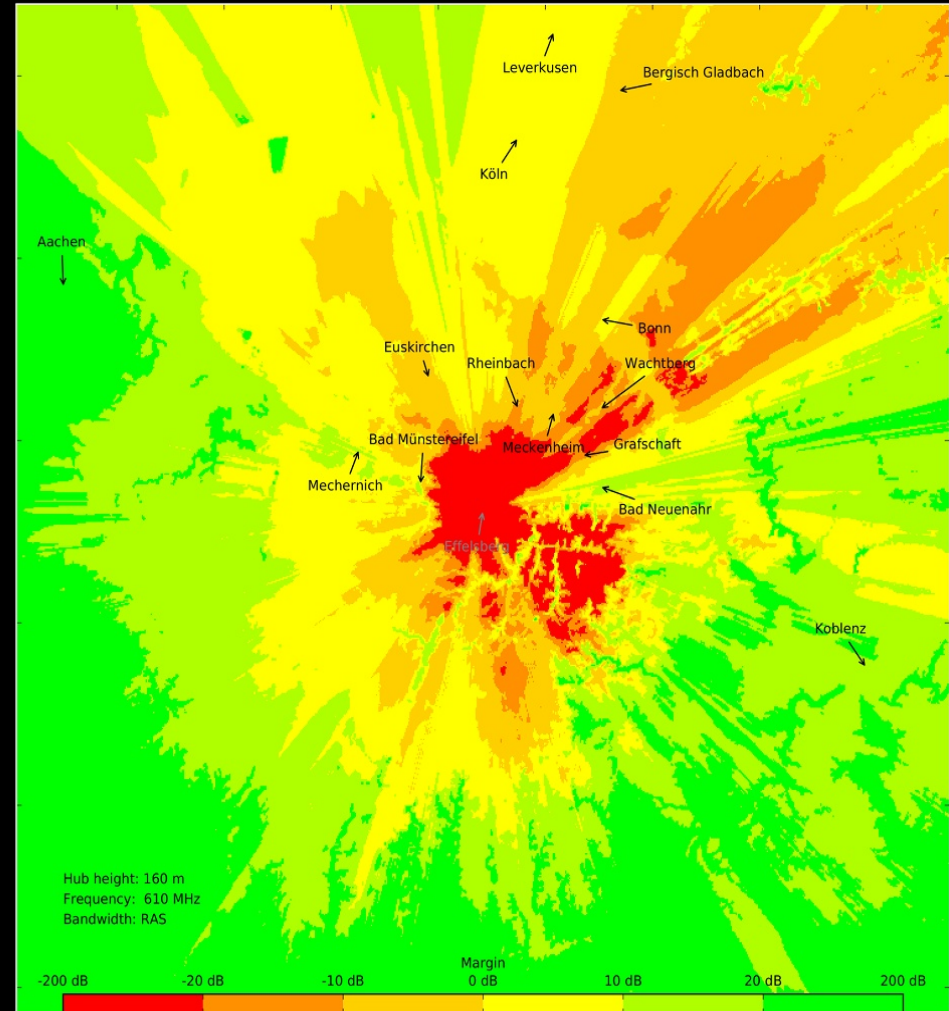
A new dark age for
radio astronomy?

Benjamin Winkel



@Hlprocessor

Max-Planck-Institut
für
Radioastronomie



Launch of the first 60 of 12000 SpaceX/Starlink satellites



Image: Marco Langbroek; May 24, 2019

A world map showing the distribution of artificial light at night. The map is dark blue, with landmasses outlined in a lighter blue. Numerous bright yellow and white dots and streaks represent city lights and urban areas. The most intense concentrations are visible in North America, Europe, and East Asia, with smaller clusters in South America, Africa, and Australia. The oceans are mostly black, indicating a lack of artificial light.

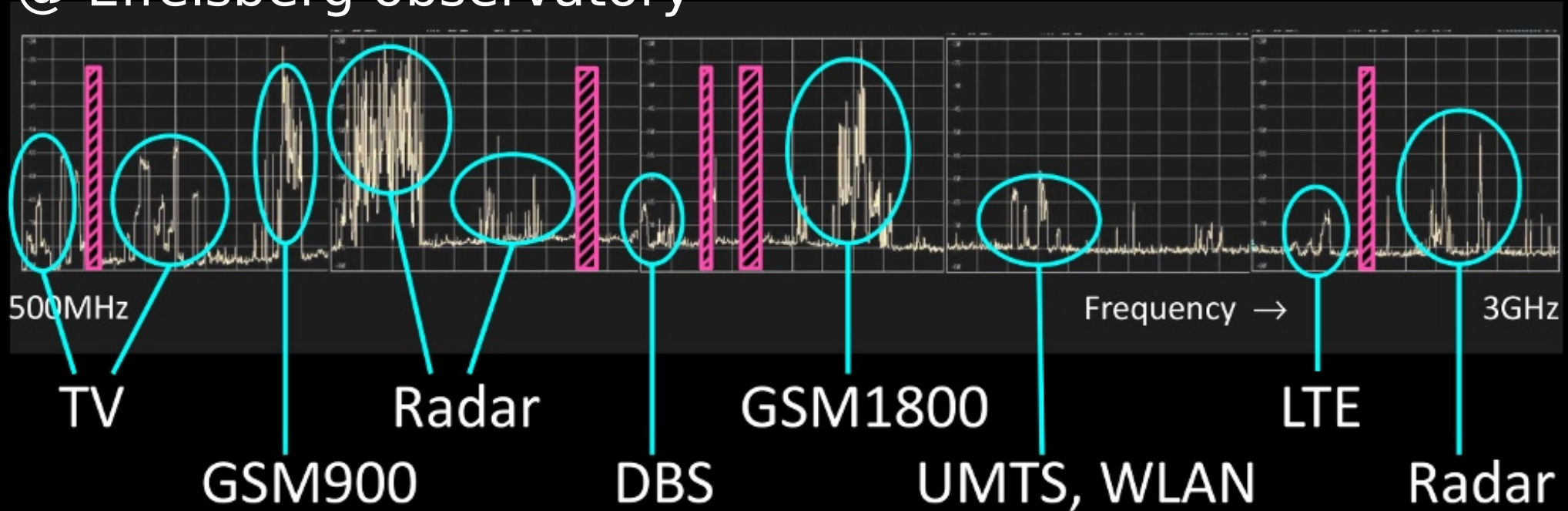
Light pollution

Image: NASA, 2016

Radio frequency interference (RFI)

@ Effelsberg observatory

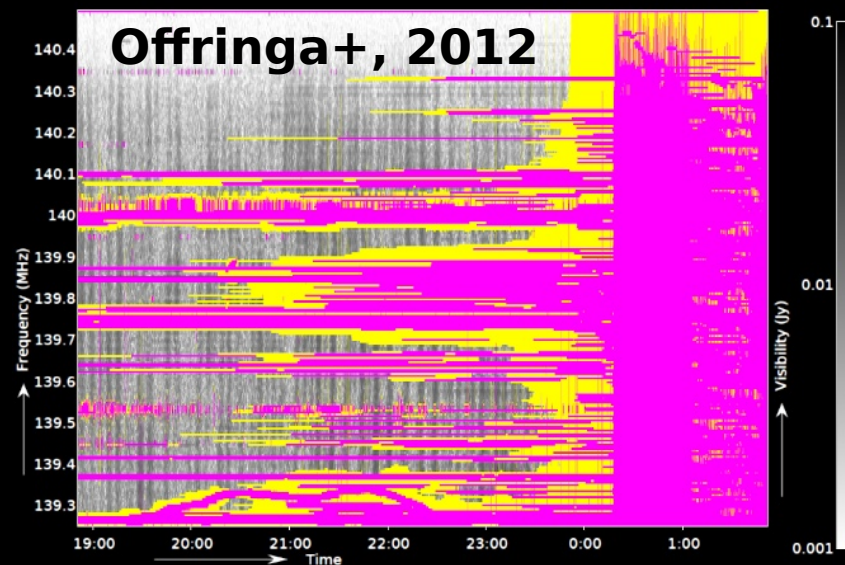
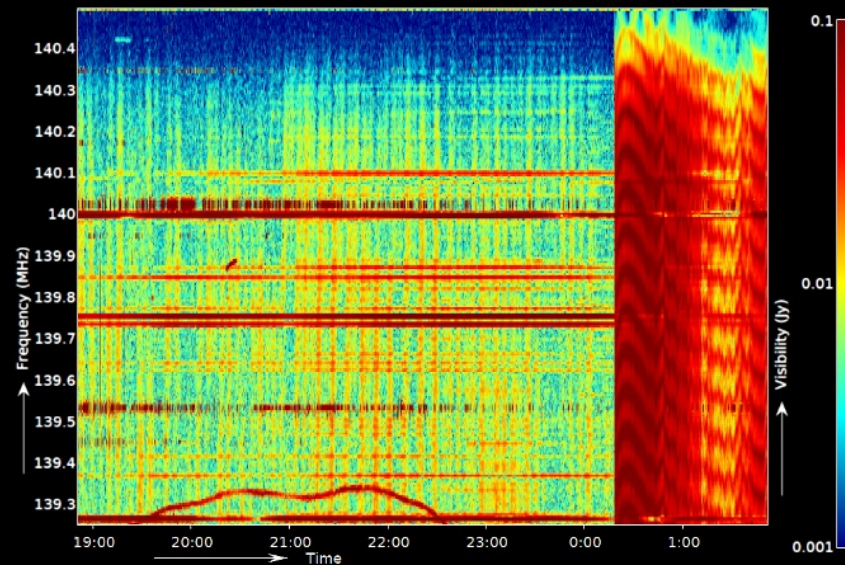
Image: MPIfR



RFI flagging or mitigation

Many tools available, e.g.

- Adaptive filtering
- Peak thresholding
- Higher order statistics
- Deep neural networks



Limits of RFI mitigation / detection

- Often highly adapted to:
 - One or few types of RFI
 - Particular observing modes
- High incident power can
 - Cause intermodulation products
 - Completely saturate receiver (blocking)
 - any kind of (digital) post-processing will fail

Limits of RFI mitigation / detection

- Often highly adapted to:
 - One or few types of RFI
 - Particular observing modes
- High incident power can
 - Cause intermodulation products
 - Completely saturate receiver (blocking)

Must limit transmit power or distance
→ Spectrum management

Spectrum management

- Lobbying :-(
- Convince administrations and companies to protect radio observatories
- **Compatibility calculations**
If you want protection, you need to proof that you are affected!

Spectrum management

- Lobbying :-(
 - Convince administrations and companies to protect radio observatories
 - **Compatibility calculations**
If you want protection, you need to proof that you are affected!



Compatibility and sharing studies related to NGSO satellite systems operating in the FSS bands 10.7-12.75 GHz (space-to-Earth) and 14-14.5 GHz (Earth-to-space)

approved 26 January 2018

Updated: 25 January 2019

Spectrum management

- Lobbying :-)
- Convince administrations and companies to protect radio observatories
- **Compatibility calculations**
If you want protection, you need to proof that you are affected!



CEPT **ECC**
Electronic Communications Committee



ECC Report **271**

Compatibility and sharing studies related to NGSO satellite systems operating in the FSS bands 10.7-12.75 GHz (space-to-Earth) and 14-14.5 GHz (Earth-to-space)

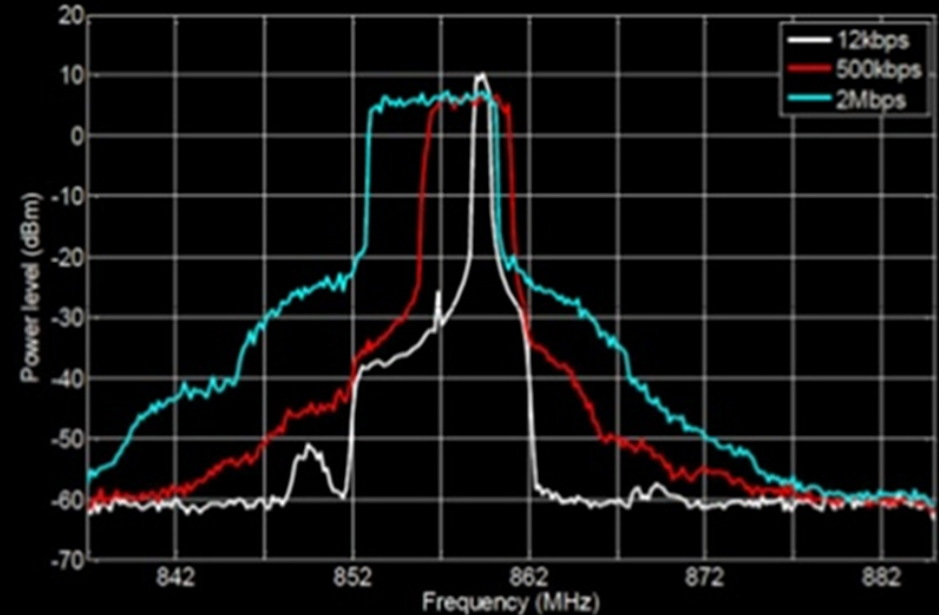
approved 26 January 2018
Updated: 25 January 2019

**Oneweb &
SpaceX/Starlink**

Compatibility calculations

Recipe

- Calculate un-/wanted emission levels of interferer
- Path propagation loss between Tx and Rx
- Infer power flux densities at victim receiver
- Compare with permitted limits

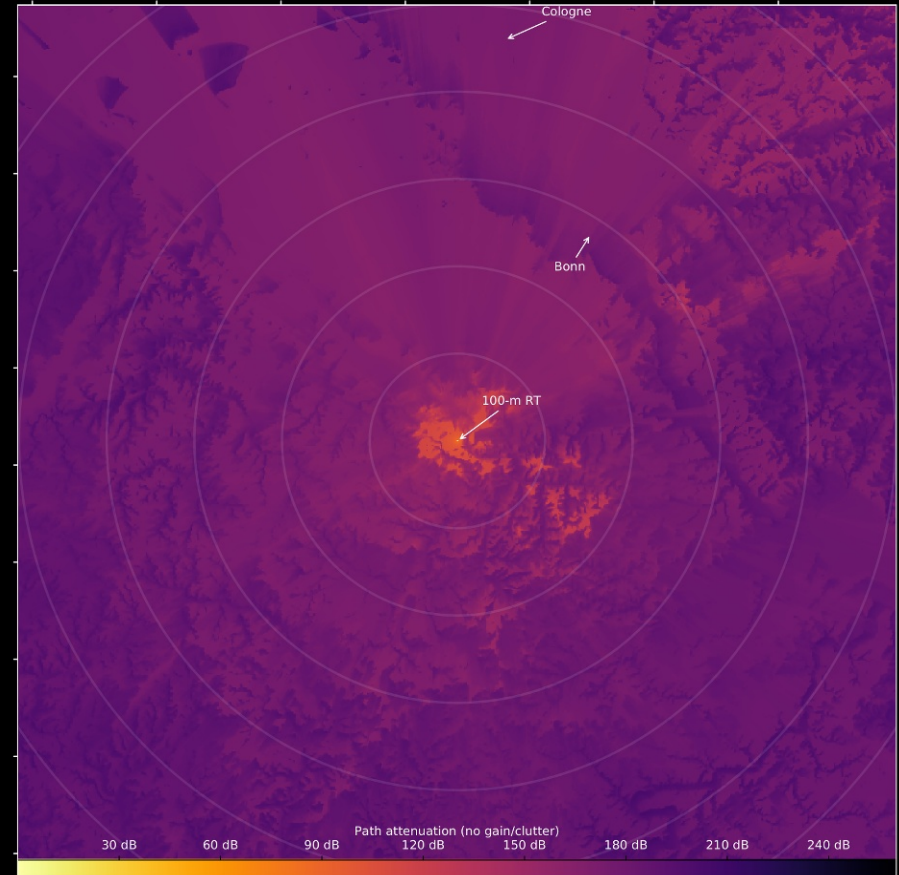


Source: Ofcom (August 2012)

Compatibility calculations

Recipe

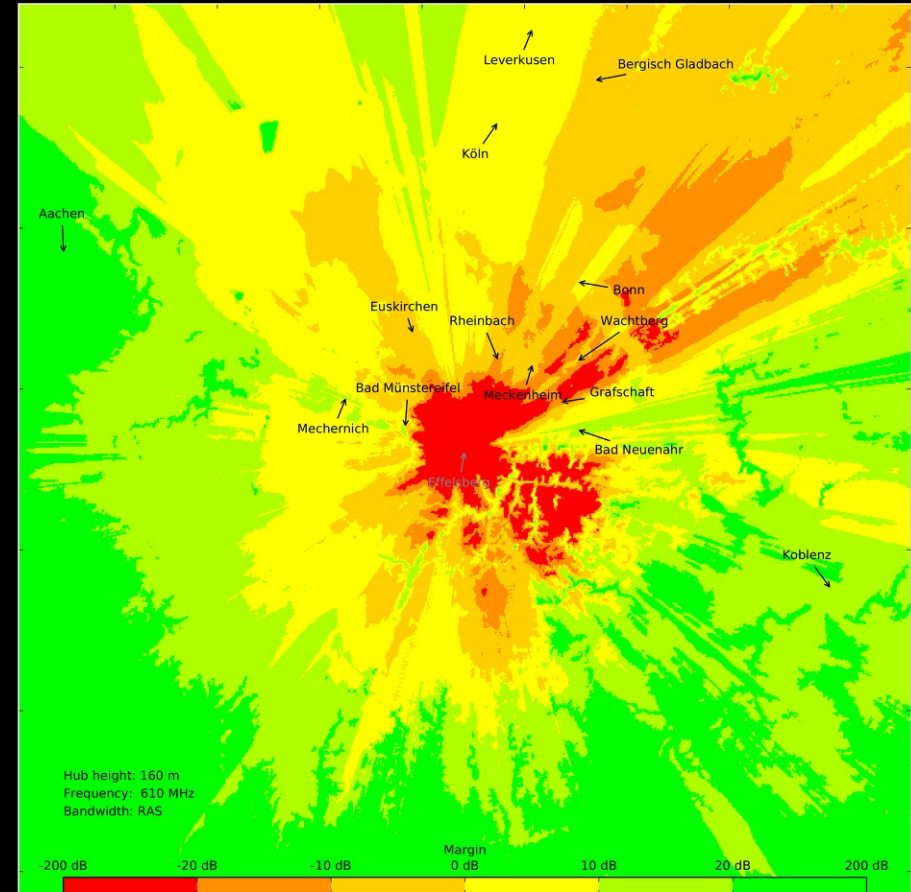
- Calculate un-/wanted emission levels of interferer
- **Path propagation loss between Tx and Rx**
- Infer power flux densities at victim receiver
- Compare with permitted limits



Compatibility calculations

Recipe

- Calculate un-/wanted emission levels of interferer
- Path propagation loss between Tx and Rx
- Infer power flux densities at victim receiver
- Compare with permitted limits



pycraf

- Python package for compatibility studies
- Winkel & Jessner 2018, 2019
- Features:
 - Path propagation loss (P.452)
 - Atmospheric attenuation (P.676)
 - Query topographical data
 - Antenna patterns
 - Geographical coordinates
 - Satellite visibility

```

heightprofile.py x conversions.py x cyprop.pyx x
241
242 @helpers.ranged_quantity_input(
243     Erx=(1.e-30, None, apu.V / apu.meter),
244     d=(1.e-30, None, apu.m),
245     Gtx=(1.e-30, None, dimless),
246     strip_input_units=True, output_unit=apu.W
247 )
248 def Ptx_from_Erx(Erx, d, Gtx):
249     ...
250     Calculate transmitter power, Ptx, from received field strength.
251
252     Note: All quantities must be astropy Quantities
253           (astropy.units.quantity.Quantity).
254
255     Parameters
256     -----
257     Erx - Received E-field strength [dB_uV_m, uV/m, or (uV/m)**2]
258     d - Distance to transmitter [m]
259     Gtx - Gain of transmitter [dBi, or dimless]
260
261     Returns
262     -----
263     Transmitter power, Ptx [W]
264     ...
265
266     return 4. * np.pi * d ** 2 / Gtx * Erx ** 2 / RO_VALUE
267
268
269 @helpers.ranged_quantity_input(
270     Ptx=(1.e-30, None, apu.W),
271     d=(1.e-30, None, apu.m),
272     Gtx=(1.e-30, None, dimless),
273     strip_input_units=True, output_unit=apu.uV / apu.meter
274 )
275 def Erx_from_Ptx(Ptx, d, Gtx):
276     ...
277     Calculate received field strength, Erx, from transmitter power.
278
279     Note: All quantities must be astropy Quantities
280           (astropy.units.quantity.Quantity).
281
282     Parameters
283     -----
284     Ptx - Transmitter power [dB_W, W]
285     d - Distance to transmitter [m]
286     Gtx - Gain of transmitter [dBi, or dimless]
287
288     Returns
289     -----
290     Received E-field strength, Erx [uV/m]
291     ...
292
293     return (Ptx * Gtx / 4. / np.pi * RO_VALUE) ** 0.5 / d * 1.e6
294
295
296 @helpers.ranged_quantity_input(
297     Ptx=(1.e-30, None, apu.W),
298     d=(1.e-30, None, apu.m),
299     Gtx=(1.e-30, None, dimless),
300     strip_input_units=True, output_unit=apu.uV / apu.meter
301 )
302 def Erx_from_Ptx_and_Gtx(Ptx, d, Gtx):
303     ...
304     Calculate received field strength, Erx, from transmitter power and gain.
305
306     Note: All quantities must be astropy Quantities
307           (astropy.units.quantity.Quantity).
308
309     Parameters
310     -----
311     Ptx - Transmitter power [dB_W, W]
312     d - Distance to transmitter [m]
313     Gtx - Gain of transmitter [dBi, or dimless]
314
315     Returns
316     -----
317     Received E-field strength, Erx [uV/m]
318     ...
319
320     return (Ptx * Gtx / 4. / np.pi * RO_VALUE) ** 0.5 / d * 1.e6
321
322
323 @helpers.ranged_quantity_input(
324     Ptx=(1.e-30, None, apu.W),
325     d=(1.e-30, None, apu.m),
326     Gtx=(1.e-30, None, dimless),
327     strip_input_units=True, output_unit=apu.uV / apu.meter
328 )
329 def Erx_from_Ptx_and_Gtx_and_d(Ptx, d, Gtx):
330     ...
331     Calculate received field strength, Erx, from transmitter power, gain, and distance.
332
333     Note: All quantities must be astropy Quantities
334           (astropy.units.quantity.Quantity).
335
336     Parameters
337     -----
338     Ptx - Transmitter power [dB_W, W]
339     d - Distance to transmitter [m]
340     Gtx - Gain of transmitter [dBi, or dimless]
341
342     Returns
343     -----
344     Received E-field strength, Erx [uV/m]
345     ...
346
347     return (Ptx * Gtx / 4. / np.pi * RO_VALUE) ** 0.5 / d * 1.e6
348
349
350 @helpers.ranged_quantity_input(
351     Ptx=(1.e-30, None, apu.W),
352     d=(1.e-30, None, apu.m),
353     Gtx=(1.e-30, None, dimless),
354     strip_input_units=True, output_unit=apu.uV / apu.meter
355 )
356 def Erx_from_Ptx_and_Gtx_and_d_and_f(Ptx, d, Gtx, f):
357     ...
358     Calculate received field strength, Erx, from transmitter power, gain, distance, and frequency.
359
360     Note: All quantities must be astropy Quantities
361           (astropy.units.quantity.Quantity).
362
363     Parameters
364     -----
365     Ptx - Transmitter power [dB_W, W]
366     d - Distance to transmitter [m]
367     Gtx - Gain of transmitter [dBi, or dimless]
368     f - Frequency [Hz]
369
370     Returns
371     -----
372     Received E-field strength, Erx [uV/m]
373     ...
374
375     return (Ptx * Gtx / 4. / np.pi * RO_VALUE) ** 0.5 / d * 1.e6
376
377
378 @helpers.ranged_quantity_input(
379     Ptx=(1.e-30, None, apu.W),
380     d=(1.e-30, None, apu.m),
381     Gtx=(1.e-30, None, dimless),
382     strip_input_units=True, output_unit=apu.uV / apu.meter
383 )
384 def Erx_from_Ptx_and_Gtx_and_d_and_f_and_theta(Ptx, d, Gtx, f, theta):
385     ...
386     Calculate received field strength, Erx, from transmitter power, gain, distance, frequency, and angle.
387
388     Note: All quantities must be astropy Quantities
389           (astropy.units.quantity.Quantity).
390
391     Parameters
392     -----
393     Ptx - Transmitter power [dB_W, W]
394     d - Distance to transmitter [m]
395     Gtx - Gain of transmitter [dBi, or dimless]
396     f - Frequency [Hz]
397     theta - Angle [radians]
398
399     Returns
400     -----
401     Received E-field strength, Erx [uV/m]
402     ...
403
404     return (Ptx * Gtx / 4. / np.pi * RO_VALUE) ** 0.5 / d * 1.e6
405
406
407 @helpers.ranged_quantity_input(
408     Ptx=(1.e-30, None, apu.W),
409     d=(1.e-30, None, apu.m),
410     Gtx=(1.e-30, None, dimless),
411     strip_input_units=True, output_unit=apu.uV / apu.meter
412 )
413 def Erx_from_Ptx_and_Gtx_and_d_and_f_and_theta_and_phi(Ptx, d, Gtx, f, theta, phi):
414     ...
415     Calculate received field strength, Erx, from transmitter power, gain, distance, frequency, angle, and azimuth.
416
417     Note: All quantities must be astropy Quantities
418           (astropy.units.quantity.Quantity).
419
420     Parameters
421     -----
422     Ptx - Transmitter power [dB_W, W]
423     d - Distance to transmitter [m]
424     Gtx - Gain of transmitter [dBi, or dimless]
425     f - Frequency [Hz]
426     theta - Angle [radians]
427     phi - Azimuth [radians]
428
429     Returns
430     -----
431     Received E-field strength, Erx [uV/m]
432     ...
433
434     return (Ptx * Gtx / 4. / np.pi * RO_VALUE) ** 0.5 / d * 1.e6
435
436
437 @helpers.ranged_quantity_input(
438     Ptx=(1.e-30, None, apu.W),
439     d=(1.e-30, None, apu.m),
440     Gtx=(1.e-30, None, dimless),
441     strip_input_units=True, output_unit=apu.uV / apu.meter
442 )
443 def Erx_from_Ptx_and_Gtx_and_d_and_f_and_theta_and_phi_and_theta_phi(Ptx, d, Gtx, f, theta, phi, theta_phi):
444     ...
445     Calculate received field strength, Erx, from transmitter power, gain, distance, frequency, angle, azimuth, and theta-phi.
446
447     Note: All quantities must be astropy Quantities
448           (astropy.units.quantity.Quantity).
449
450     Parameters
451     -----
452     Ptx - Transmitter power [dB_W, W]
453     d - Distance to transmitter [m]
454     Gtx - Gain of transmitter [dBi, or dimless]
455     f - Frequency [Hz]
456     theta - Angle [radians]
457     phi - Azimuth [radians]
458     theta_phi - Theta-phi [radians]
459
460     Returns
461     -----
462     Received E-field strength, Erx [uV/m]
463     ...
464
465     return (Ptx * Gtx / 4. / np.pi * RO_VALUE) ** 0.5 / d * 1.e6
466
467
468 @helpers.ranged_quantity_input(
469     Ptx=(1.e-30, None, apu.W),
470     d=(1.e-30, None, apu.m),
471     Gtx=(1.e-30, None, dimless),
472     strip_input_units=True, output_unit=apu.uV / apu.meter
473 )
474 def Erx_from_Ptx_and_Gtx_and_d_and_f_and_theta_and_phi_and_theta_phi_and_theta_phi_phi(Ptx, d, Gtx, f, theta, phi, theta_phi, theta_phi_phi):
475     ...
476     Calculate received field strength, Erx, from transmitter power, gain, distance, frequency, angle, azimuth, theta-phi, and theta-phi-phi.
477
478     Note: All quantities must be astropy Quantities
479           (astropy.units.quantity.Quantity).
480
481     Parameters
482     -----
483     Ptx - Transmitter power [dB_W, W]
484     d - Distance to transmitter [m]
485     Gtx - Gain of transmitter [dBi, or dimless]
486     f - Frequency [Hz]
487     theta - Angle [radians]
488     phi - Azimuth [radians]
489     theta_phi - Theta-phi [radians]
490     theta_phi_phi - Theta-phi-phi [radians]
491
492     Returns
493     -----
494     Received E-field strength, Erx [uV/m]
495     ...
496
497     return (Ptx * Gtx / 4. / np.pi * RO_VALUE) ** 0.5 / d * 1.e6
498
499
500 @helpers.ranged_quantity_input(
501     Ptx=(1.e-30, None, apu.W),
502     d=(1.e-30, None, apu.m),
503     Gtx=(1.e-30, None, dimless),
504     strip_input_units=True, output_unit=apu.uV / apu.meter
505 )
506 def Erx_from_Ptx_and_Gtx_and_d_and_f_and_theta_and_phi_and_theta_phi_and_theta_phi_phi_and_theta_phi_phi_phi(Ptx, d, Gtx, f, theta, phi, theta_phi, theta_phi_phi, theta_phi_phi_phi):
507     ...
508     Calculate received field strength, Erx, from transmitter power, gain, distance, frequency, angle, azimuth, theta-phi, theta-phi-phi, and theta-phi-phi-phi.
509
510     Note: All quantities must be astropy Quantities
511           (astropy.units.quantity.Quantity).
512
513     Parameters
514     -----
515     Ptx - Transmitter power [dB_W, W]
516     d - Distance to transmitter [m]
517     Gtx - Gain of transmitter [dBi, or dimless]
518     f - Frequency [Hz]
519     theta - Angle [radians]
520     phi - Azimuth [radians]
521     theta_phi - Theta-phi [radians]
522     theta_phi_phi - Theta-phi-phi [radians]
523     theta_phi_phi_phi - Theta-phi-phi-phi [radians]
524
525     Returns
526     -----
527     Received E-field strength, Erx [uV/m]
528     ...
529
530     return (Ptx * Gtx / 4. / np.pi * RO_VALUE) ** 0.5 / d * 1.e6
531
532
533 @helpers.ranged_quantity_input(
534     Ptx=(1.e-30, None, apu.W),
535     d=(1.e-30, None, apu.m),
536     Gtx=(1.e-30, None, dimless),
537     strip_input_units=True, output_unit=apu.uV / apu.meter
538 )
539 def Erx_from_Ptx_and_Gtx_and_d_and_f_and_theta_and_phi_and_theta_phi_and_theta_phi_phi_and_theta_phi_phi_phi_and_theta_phi_phi_phi_phi(Ptx, d, Gtx, f, theta, phi, theta_phi, theta_phi_phi, theta_phi_phi_phi, theta_phi_phi_phi_phi):
540     ...
541     Calculate received field strength, Erx, from transmitter power, gain, distance, frequency, angle, azimuth, theta-phi, theta-phi-phi, theta-phi-phi-phi, and theta-phi-phi-phi-phi.
542
543     Note: All quantities must be astropy Quantities
544           (astropy.units.quantity.Quantity).
545
546     Parameters
547     -----
548     Ptx - Transmitter power [dB_W, W]
549     d - Distance to transmitter [m]
550     Gtx - Gain of transmitter [dBi, or dimless]
551     f - Frequency [Hz]
552     theta - Angle [radians]
553     phi - Azimuth [radians]
554     theta_phi - Theta-phi [r
```


pycraf

- Free & OSS
- Hosted on GitHub
- Powered by Astropy package template
- Well documented
- Many examples
- Contributions welcome!
- `pip install pycraf`
- `conda install pycraf`
(conda-forge channel)

<https://github.com/bwinkel/pycraf>

pycraf

- *Version:* 0.25
- *Author:* Benjamin Winkel
- *User manual:* [stable](#) | [developer](#)

`pypi` `v0.25.8` `license` `GPL` `DOI` `10.5281/zenodo.1244192`

The pycraf Python package provides functions and procedures for various tasks in spectrum-management compatibility studies. A typical example would be to calculate the interference levels at a radio telescope produced from a radio broadcasting tower.

Releases are [registered on PyPI](#), and development is occurring at the [project's github page](#).

Project Status

`build` `passing` `build` `passing` `coverage` `92%`

pycraf is still in the early-development stage. While much of the functionality is already working as intended, the API is not yet stable. Nevertheless, we kindly invite you to use and test the library and we are grateful for feedback. Note, that work on the documentation is still ongoing.

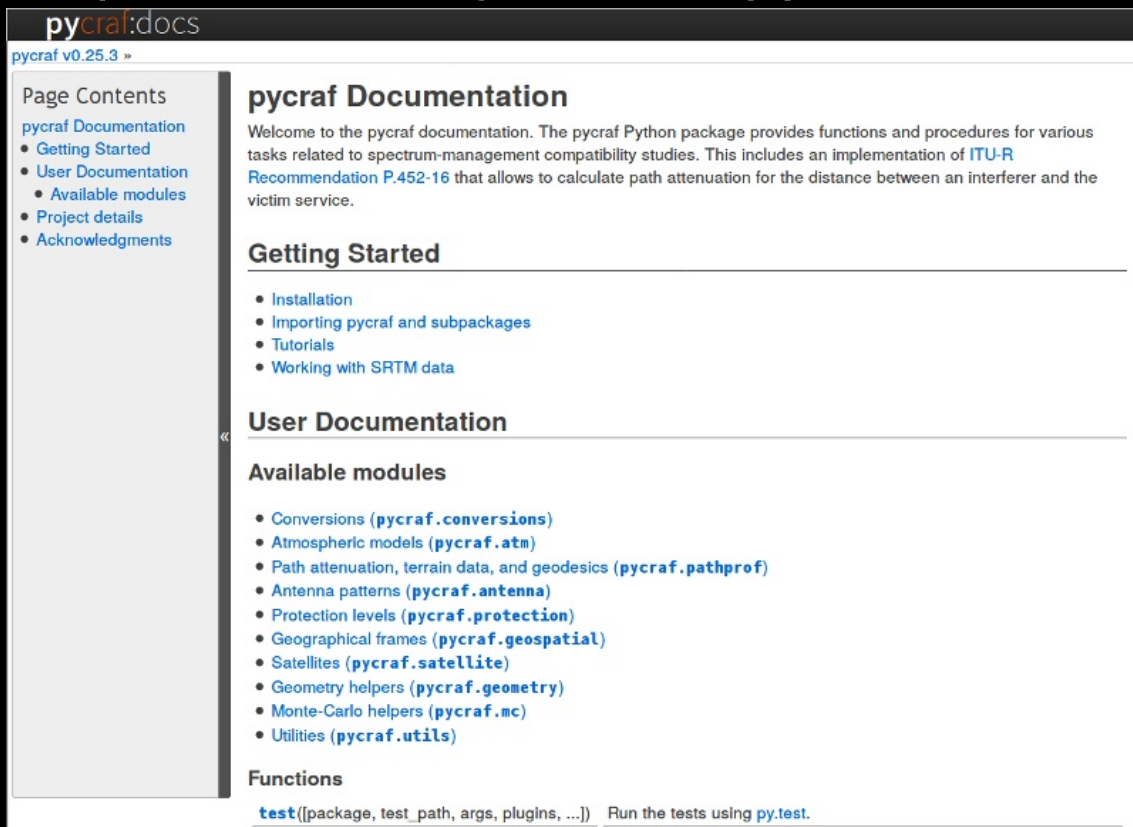
Features

- Full implementation of [ITU-R Rec. P.452-16](#) that allows to calculate path attenuation for the distance between interferer and victim service. Supports to load NASA's [Shuttle Radar Topography Mission \(SRTM\)](#) data for height-profile generation.
- Full implementation of [ITU-R Rec. P.676-10](#), which provides two atmospheric models to calculate the attenuation for paths through Earth's atmosphere.
- Provides various antenna patterns necessary for compatibility studies (e.g., RAS, IMT, fixed-service links).
- Functions to convert power flux densities, field strengths, transmitted and received powers at certain distances and frequencies into each other.

pycraf

- Free & OSS
- Hosted on GitHub
- Powered by Astropy package template
- Well documented
- Many examples
- Contributions welcome!
- `pip install pycraf`
- `conda install pycraf`
(conda-forge channel)

<https://bwinkel.github.io/pycraf/>



The screenshot shows the pycraf documentation website. The header includes the title 'pycraf:docs' and the version 'pycraf v0.25.3'. A left sidebar contains a 'Page Contents' menu with links to 'pycraf Documentation', 'Getting Started', 'User Documentation', 'Available modules', 'Project details', and 'Acknowledgments'. The main content area is titled 'pycraf Documentation' and contains a welcome message, a 'Getting Started' section with links to 'Installation', 'Importing pycraf and subpackages', 'Tutorials', and 'Working with SRTM data', a 'User Documentation' section, an 'Available modules' section listing various submodules like 'Conversions', 'Atmospheric models', 'Path attenuation', etc., and a 'Functions' section with a link to 'test'.

pycraf:docs
pycraf v0.25.3 »

Page Contents

- pycraf Documentation
- [Getting Started](#)
- [User Documentation](#)
- [Available modules](#)
- [Project details](#)
- [Acknowledgments](#)

pycraf Documentation

Welcome to the pycraf documentation. The pycraf Python package provides functions and procedures for various tasks related to spectrum-management compatibility studies. This includes an implementation of [ITU-R Recommendation P.452-16](#) that allows to calculate path attenuation for the distance between an interferer and the victim service.

Getting Started

- [Installation](#)
- [Importing pycraf and subpackages](#)
- [Tutorials](#)
- [Working with SRTM data](#)

User Documentation

Available modules

- [Conversions](#) ([pycraf.conversions](#))
- [Atmospheric models](#) ([pycraf.atm](#))
- [Path attenuation, terrain data, and geodesics](#) ([pycraf.pathprof](#))
- [Antenna patterns](#) ([pycraf.antenna](#))
- [Protection levels](#) ([pycraf.protection](#))
- [Geographical frames](#) ([pycraf.geospatial](#))
- [Satellites](#) ([pycraf.satellite](#))
- [Geometry helpers](#) ([pycraf.geometry](#))
- [Monte-Carlo helpers](#) ([pycraf.mc](#))
- [Utilities](#) ([pycraf.utils](#))

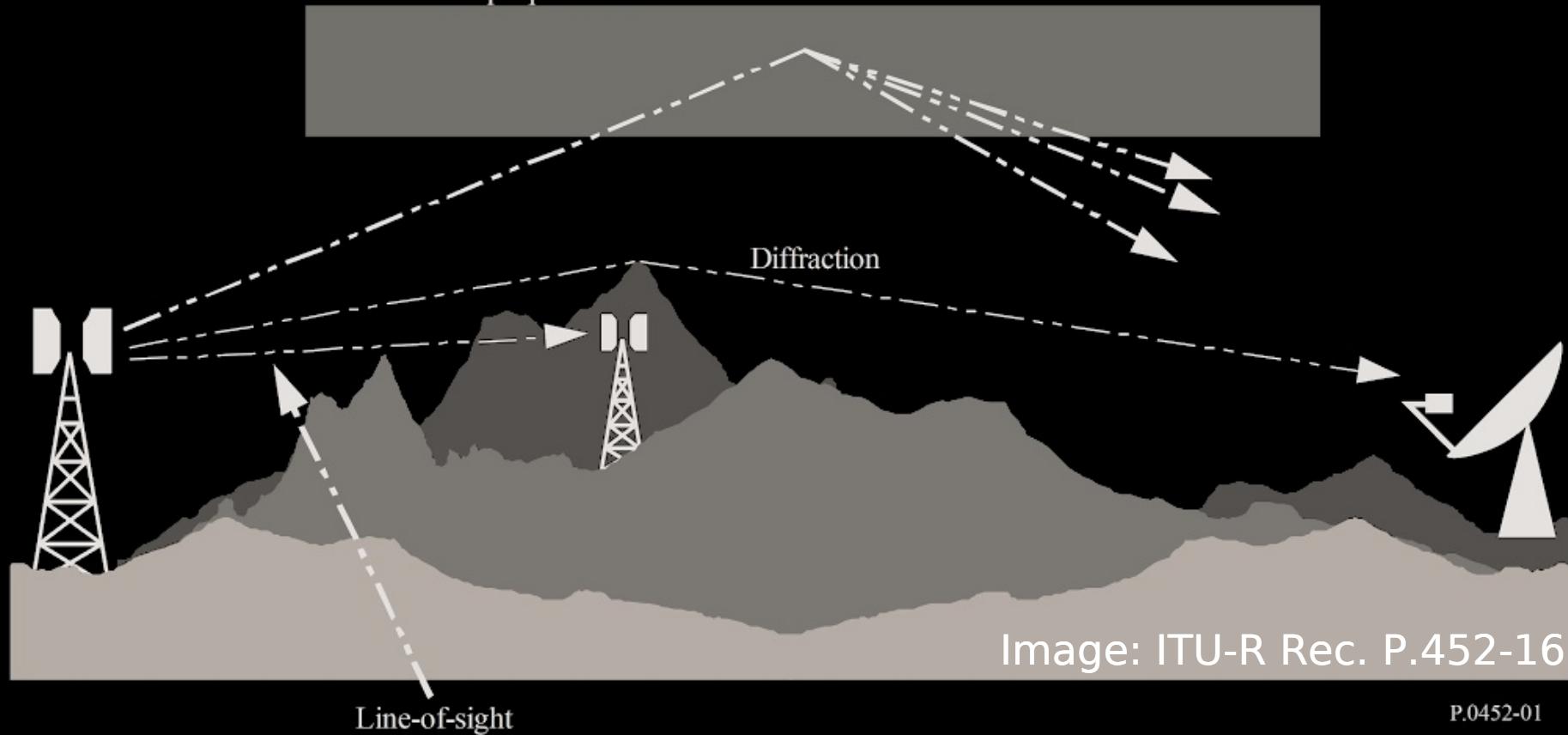
Functions

[test](#)([package, test_path, args, plugins, ...]) Run the tests using [py.test](#).

Path propagation loss

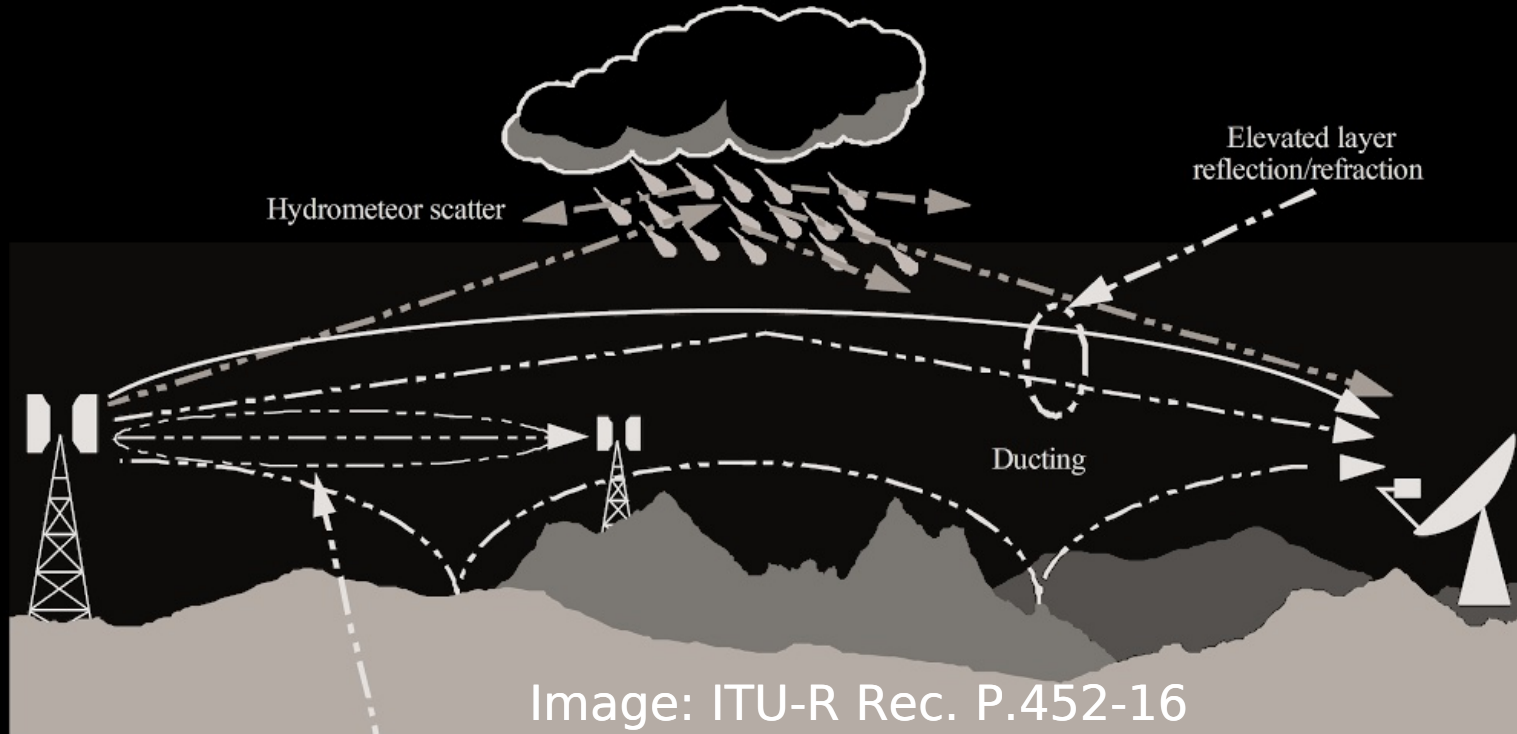
Long-term interference propagation mechanisms

Tropospheric scatter



Path propagation loss

Anomalous (short-term) interference propagation mechanisms



Path propagation loss

Knife edge diffraction

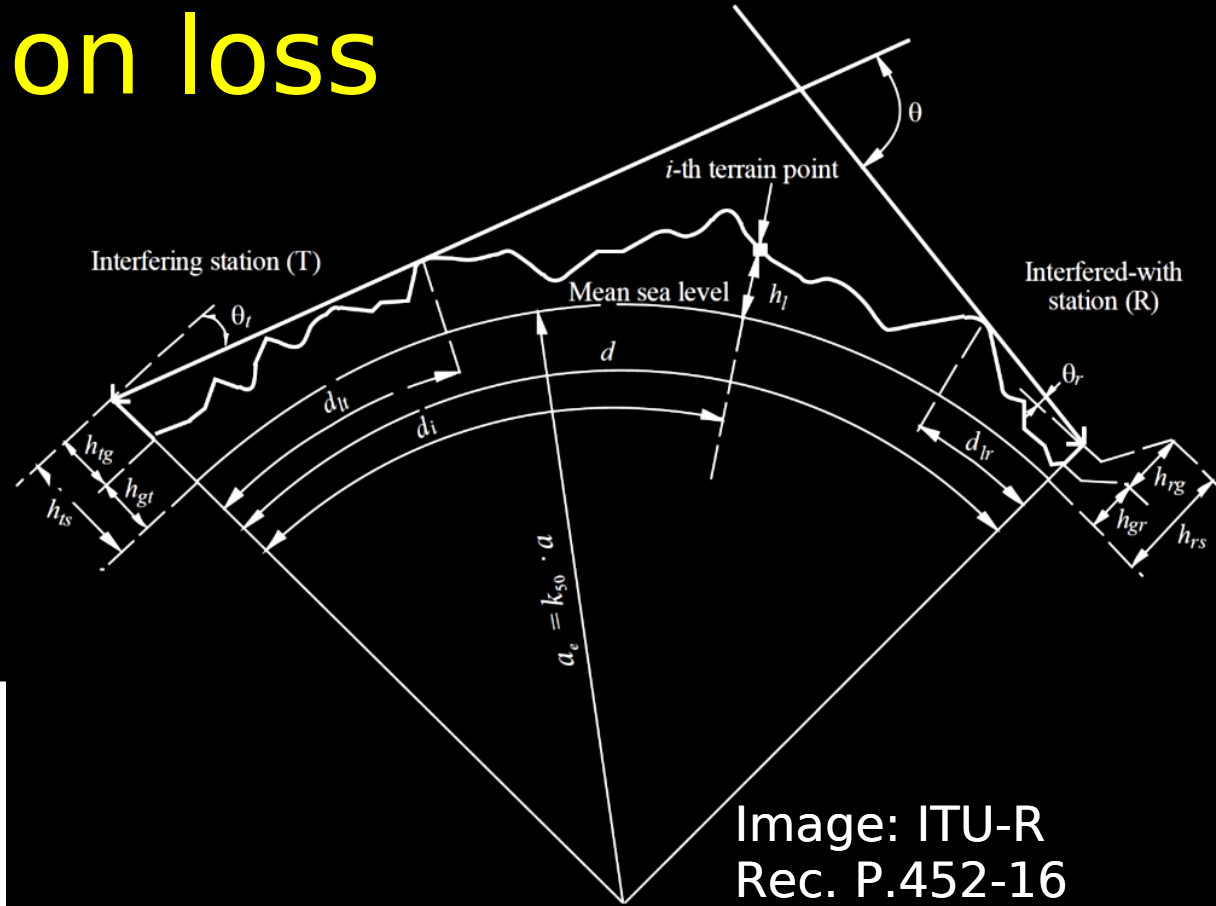
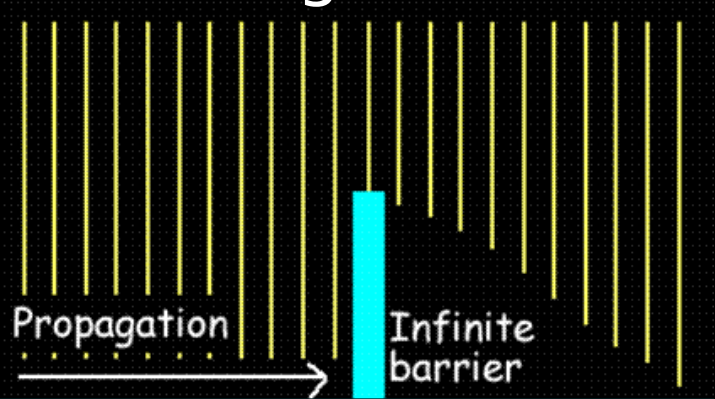
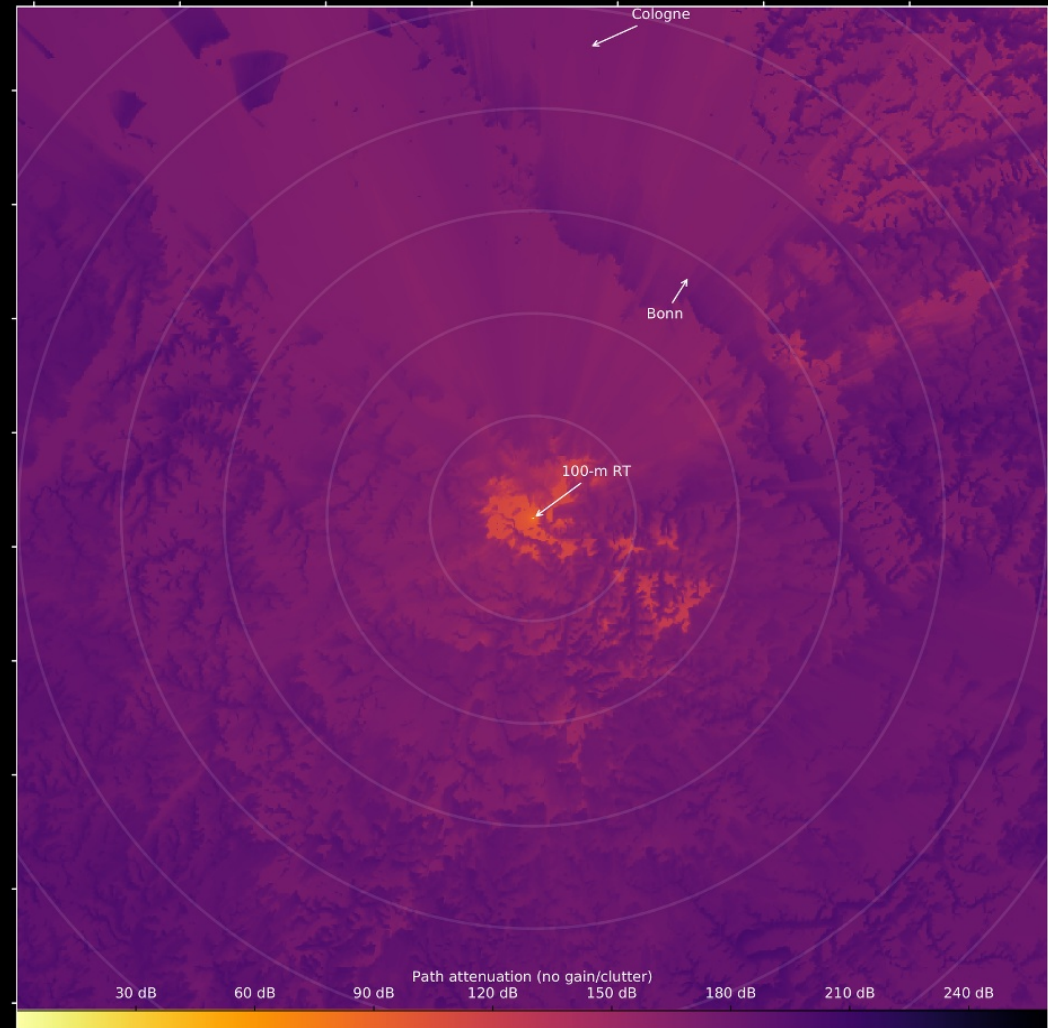


Image: ITU-R
Rec. P.452-16

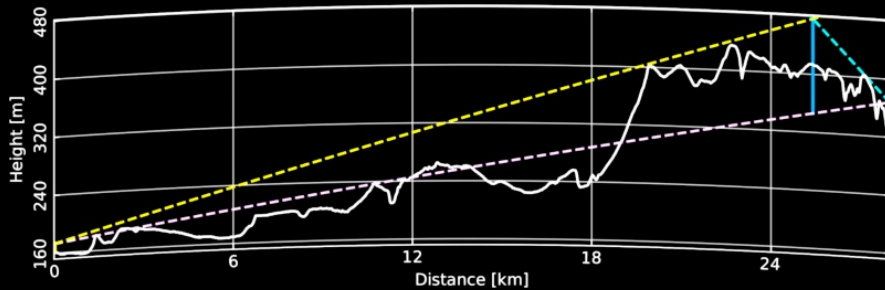
Images: Mike Willis

Propagation loss maps

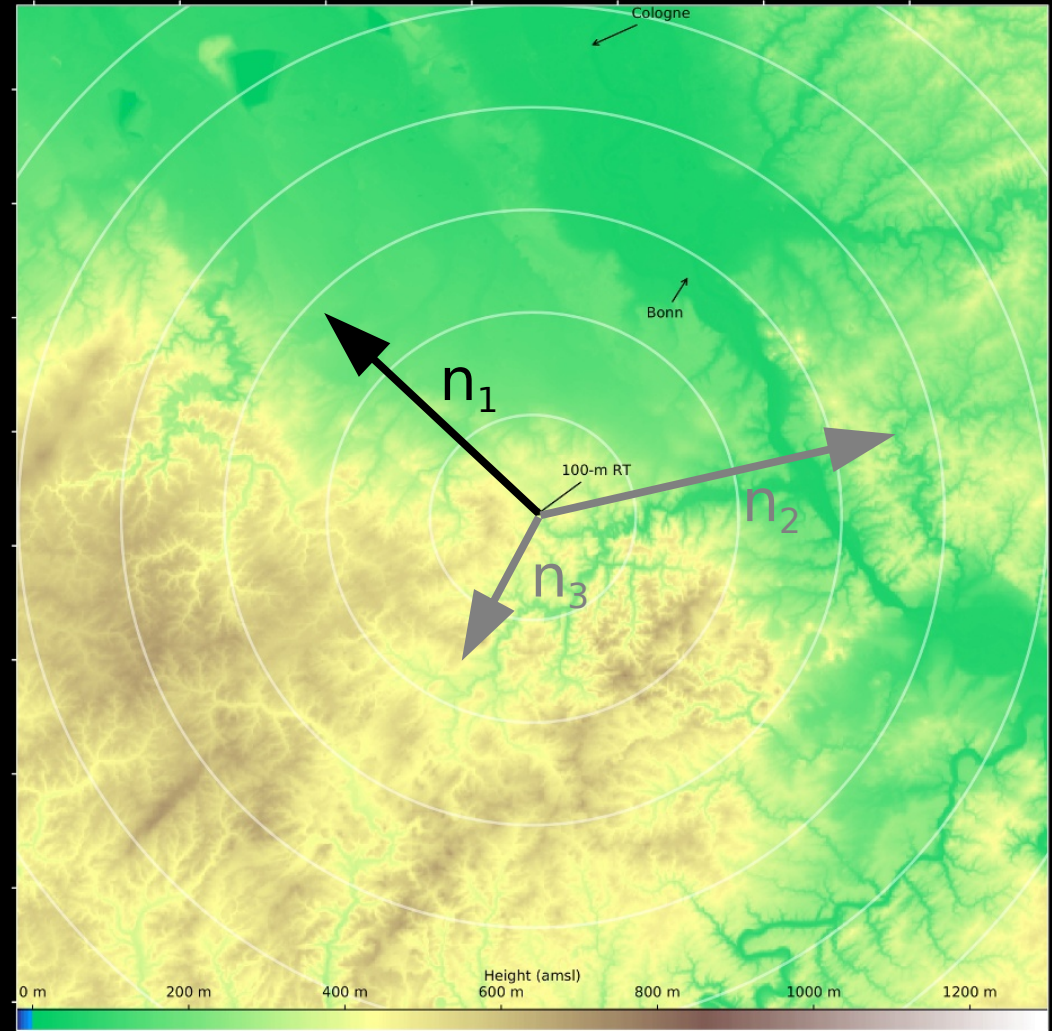


Propagation loss maps

- Query height profiles
- $O(n^3)$ problem



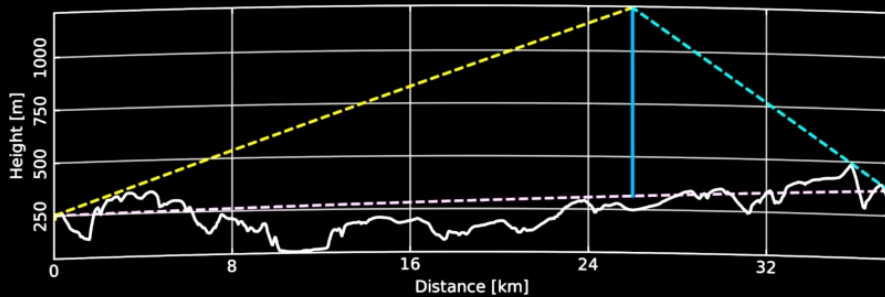
n



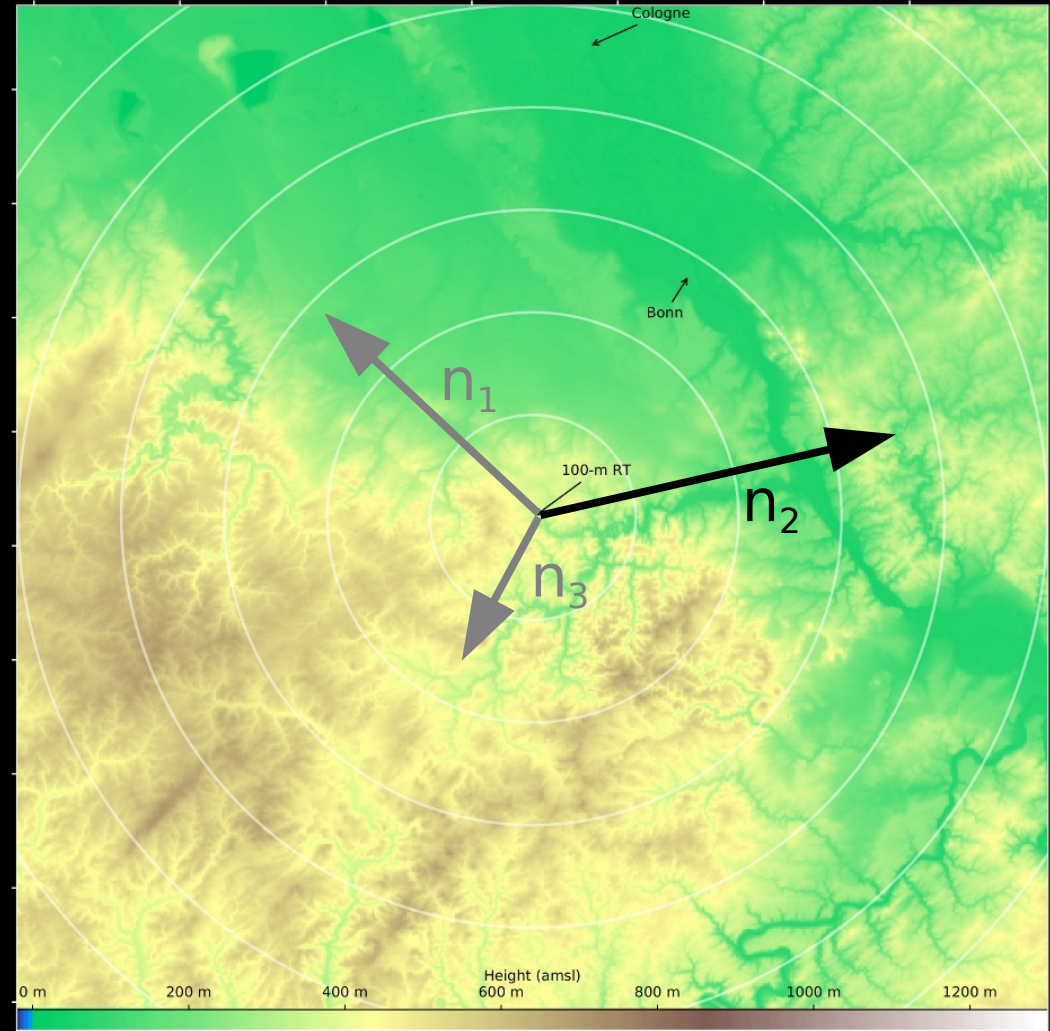
n

Propagation loss maps

- Query height profiles
- $O(n^3)$ problem



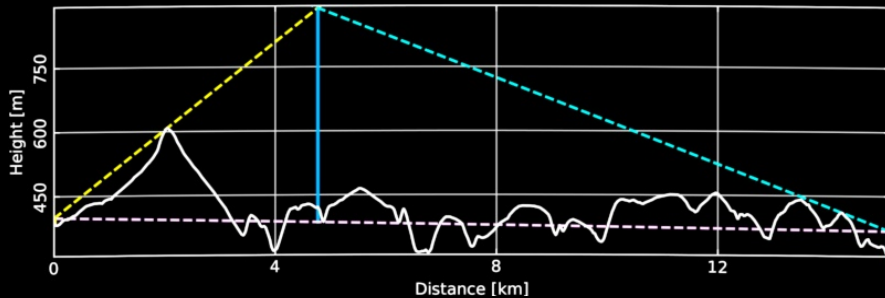
n



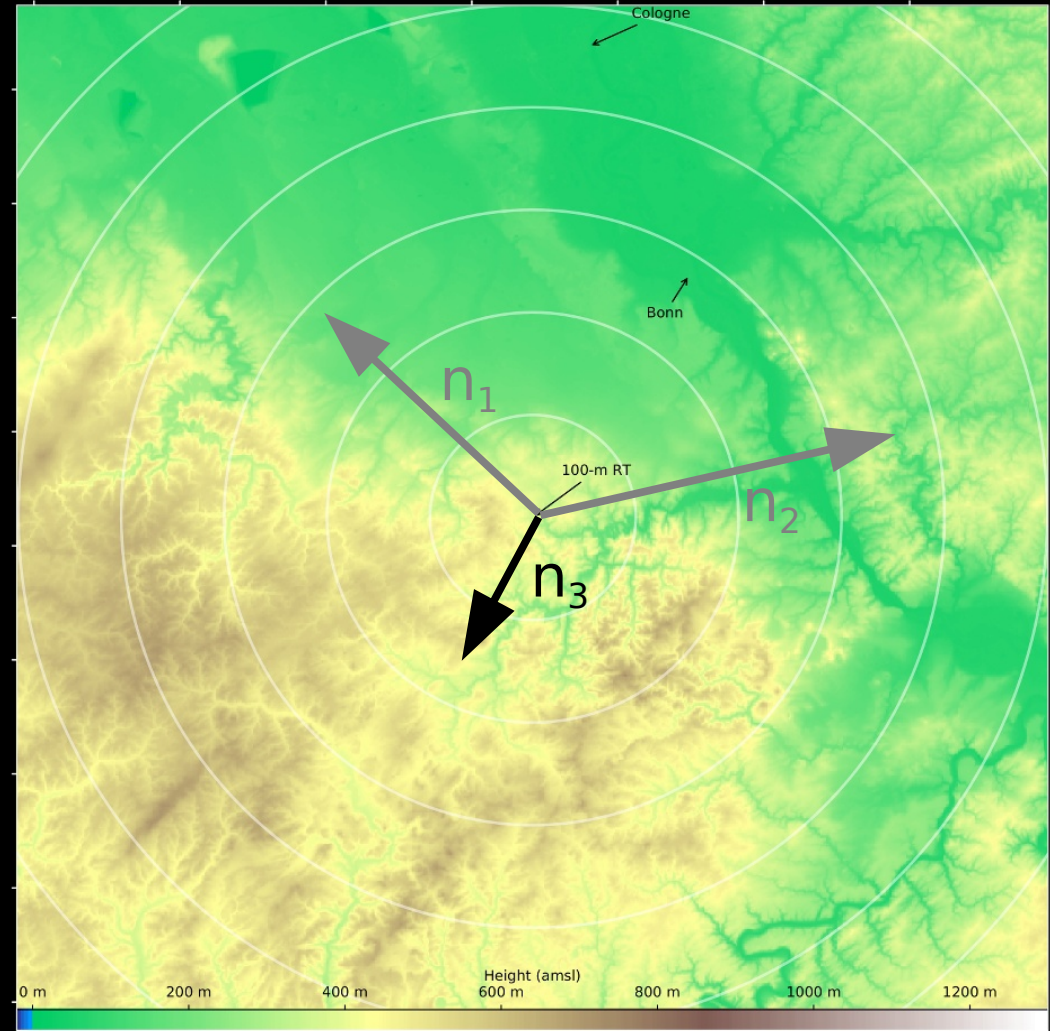
n

Propagation loss maps

- Query height profiles
- $O(n^3)$ problem



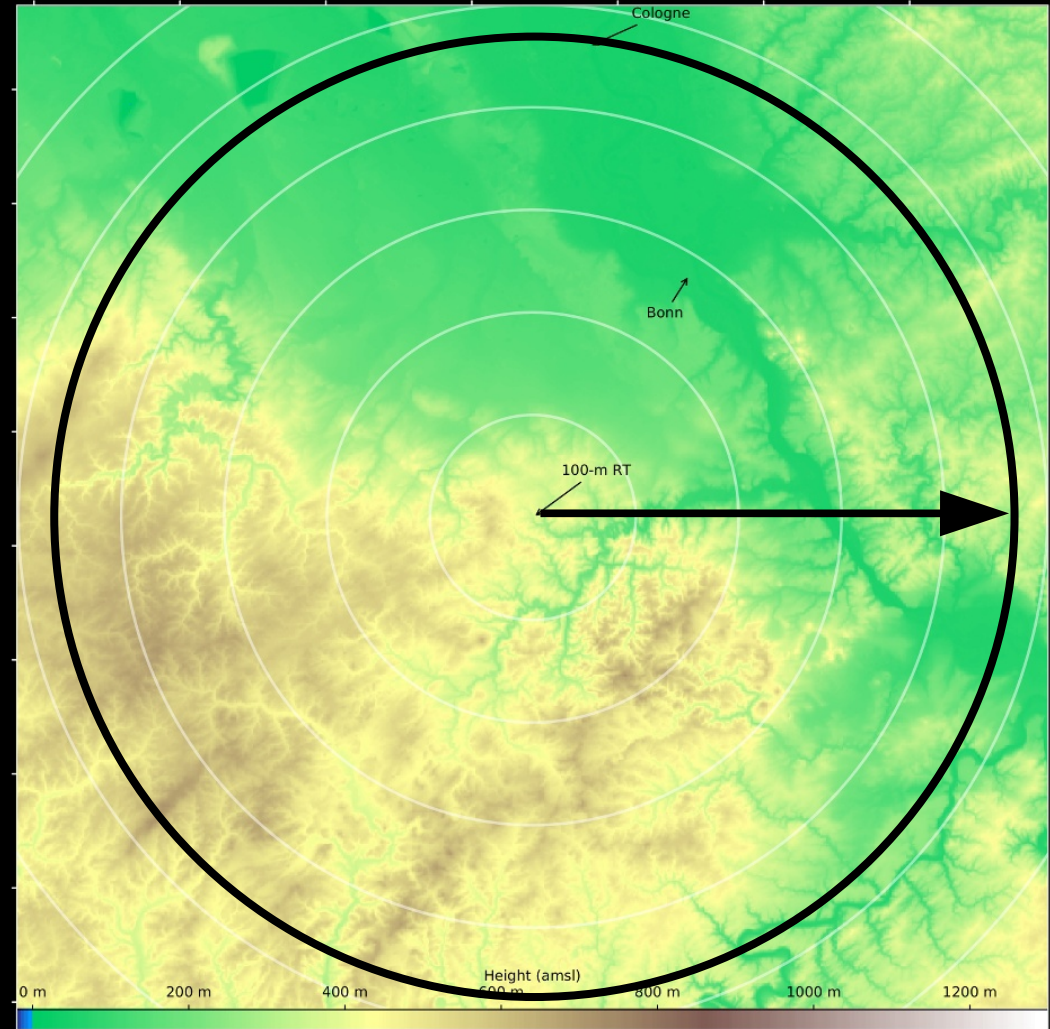
n



n

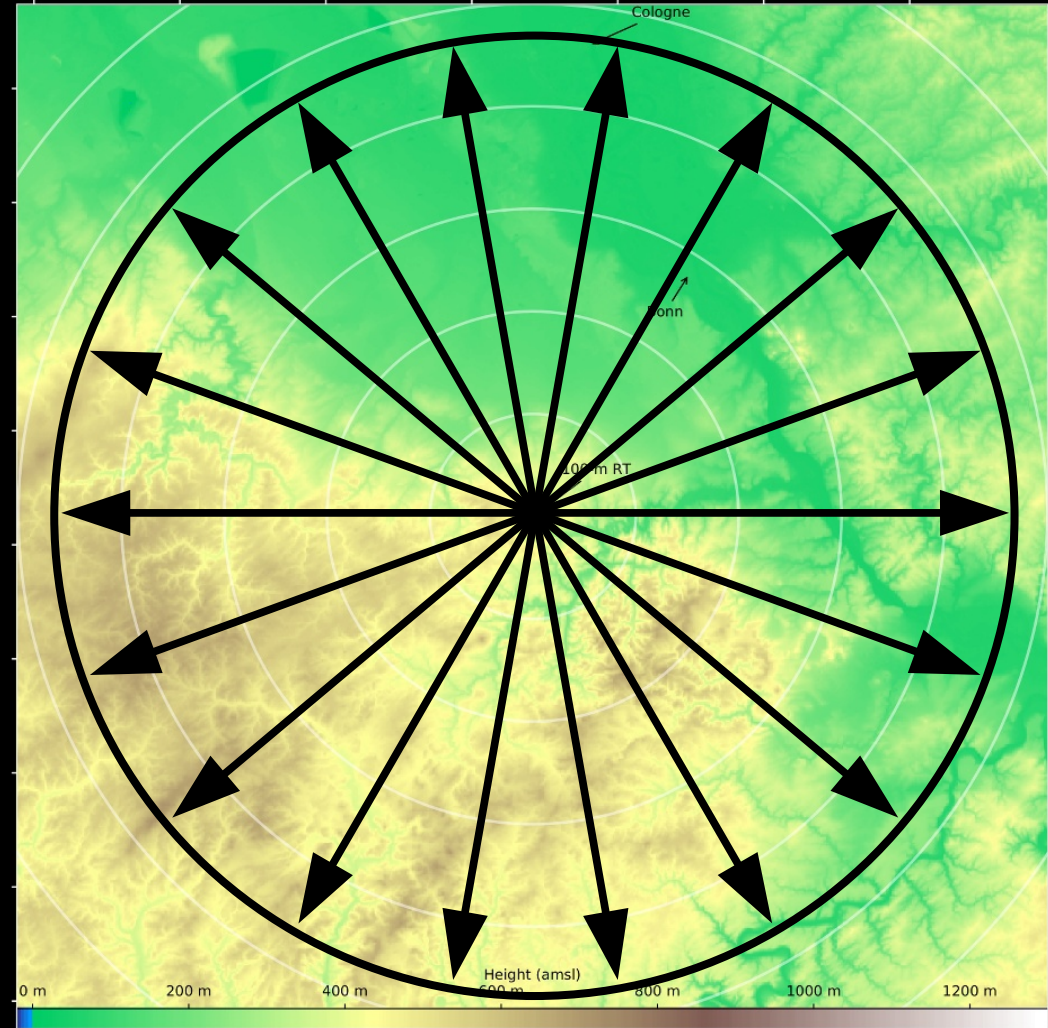
Propagation loss maps

- Query height profiles
- $O(n^3)$ problem
- Memoization



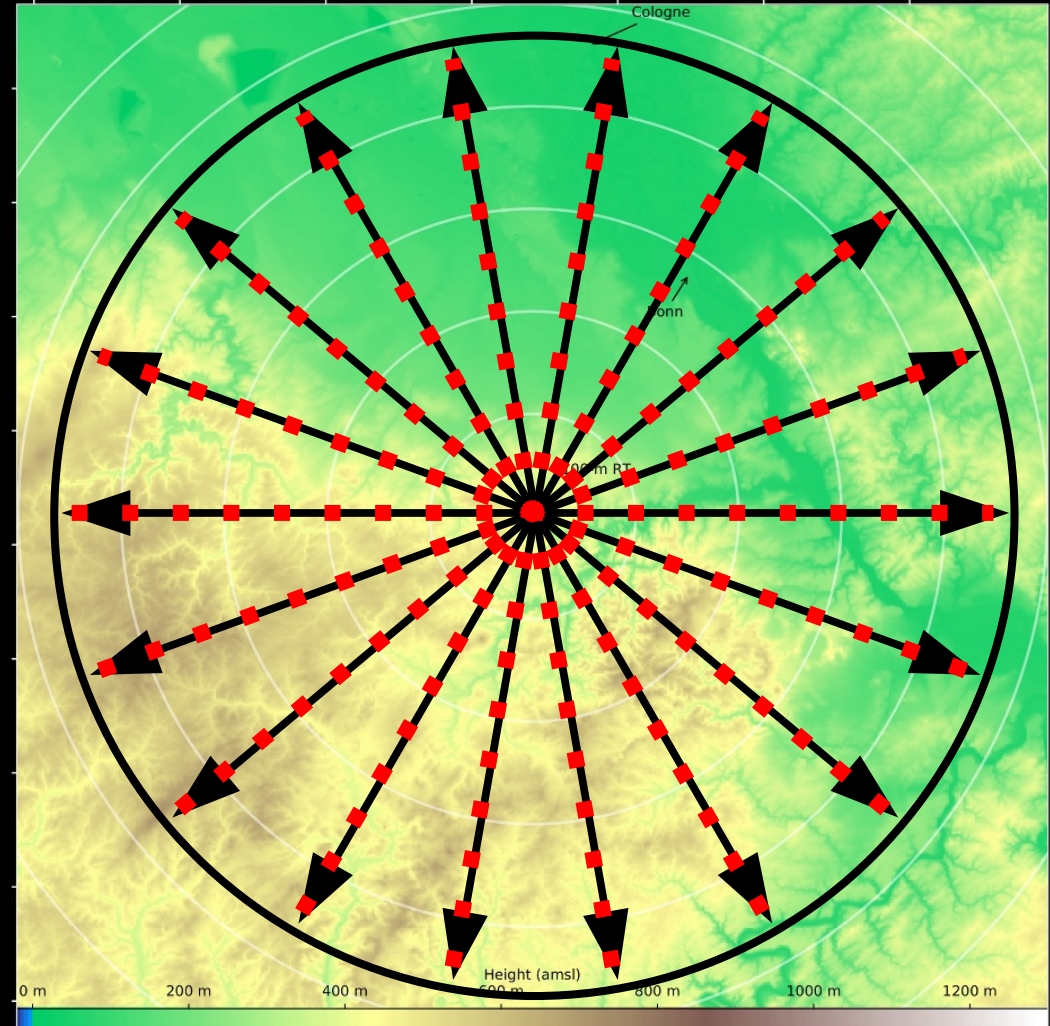
Propagation loss maps

- Query height profiles
- $O(n^3)$ problem
- Memoization



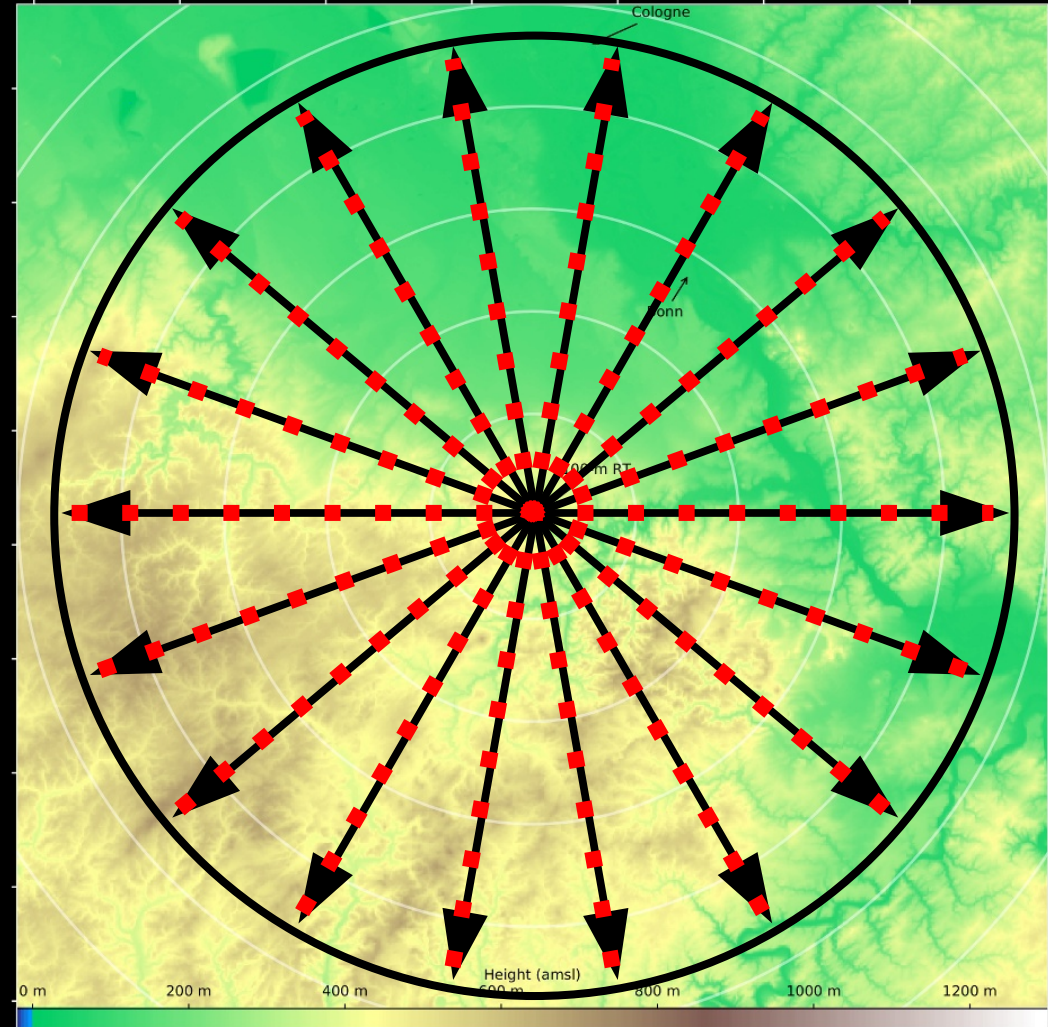
Propagation loss maps

- Query height profiles
- $O(n^3) \rightarrow O(n^2)$ problem
- Memoization
→ Look-up table
(can be stored on disk)



Propagation loss maps

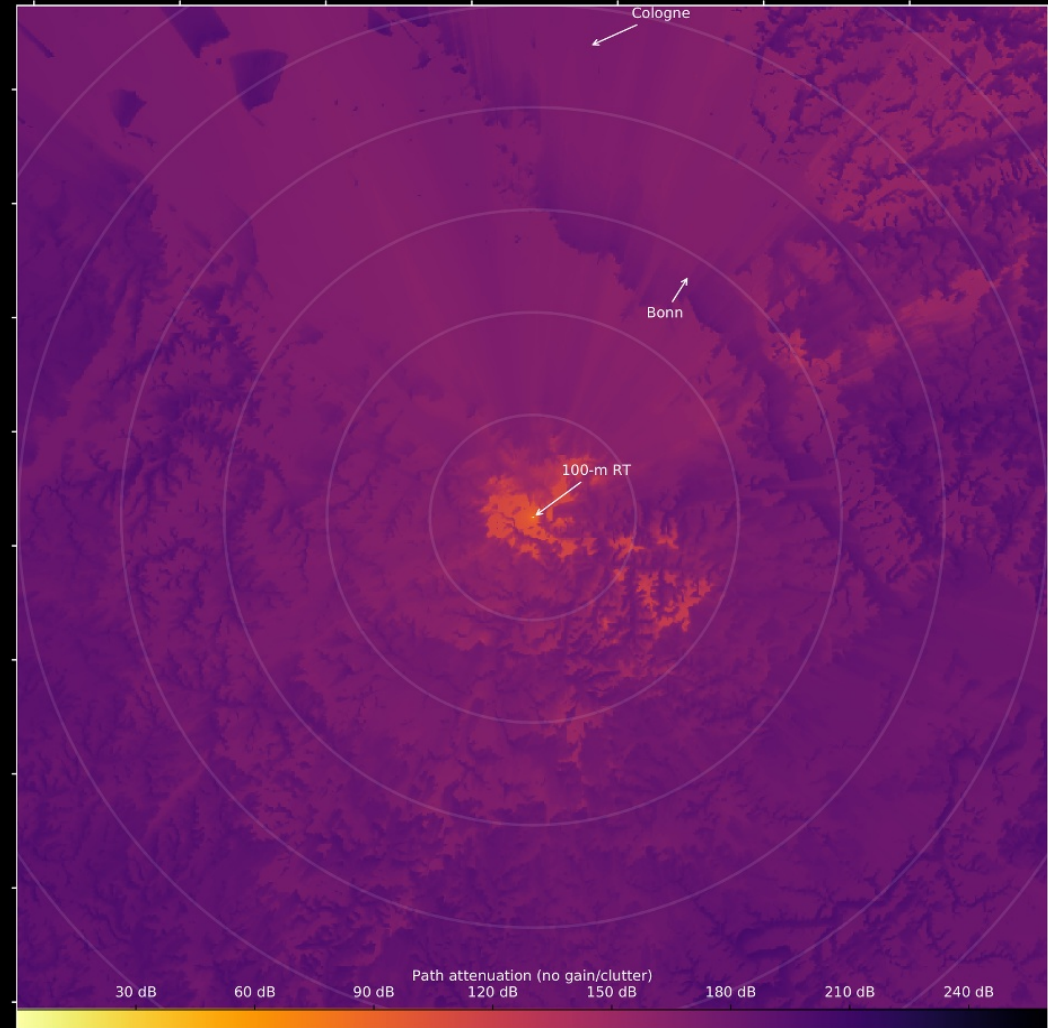
- Query height profiles
 - $O(n^3) \rightarrow O(n^2)$ problem
 - Memoization
 - Parallelization
- Cython



Propagation loss maps

- Query height profiles
- $O(n^3) \rightarrow O(n^2)$ problem
- Memoization
- Parallelization
- Cython

1–2 dex faster than other implementations



Conclusions

- Spectrum management is vital for radio astronomy
- Python package, “pycraf”, for compatibility studies
- First optimize algorithms, then code

Backup slides

World Radiocomm. Conference

- Updates radio regulations
 - Frequency table
 - Footnotes
 - Procedures
- Meets every 3–4 yrs
- Decides on “Agenda Items”
(for next WRC)
 - Als are studied in working groups
 - Changing RRs needs ≥ 5 years



Allocation tables

ITU-R Radio Regulations 2016

10.7-11.7 GHz		
Allocation to services		
Region 1	Region 2	Region 3
10.7-10.95 FIXED FIXED-SATELLITE (space-to-Earth) 5.441 (Earth-to-space) 5.484 MOBILE except aeronautical mobile	10.7-10.95 FIXED FIXED-SATELLITE (space-to-Earth) 5.441 MOBILE except aeronautical mobile	
10.95-11.2 FIXED FIXED-SATELLITE (space-to-Earth) 5.484A 5.484B (Earth-to-space) 5.484 MOBILE except aeronautical mobile	10.95-11.2 FIXED FIXED-SATELLITE (space-to-Earth) 5.484A 5.484B MOBILE except aeronautical mobile	
11.2-11.45 FIXED FIXED-SATELLITE (space-to-Earth) 5.441 (Earth-to-space) 5.484 MOBILE except aeronautical mobile	11.2-11.45 FIXED FIXED-SATELLITE (space-to-Earth) 5.441 MOBILE except aeronautical mobile	
11.45-11.7 FIXED FIXED-SATELLITE (space-to-Earth) 5.484A 5.484B (Earth-to-space) 5.484 MOBILE except aeronautical mobile	11.45-11.7 FIXED FIXED-SATELLITE (space-to-Earth) 5.484A 5.484B MOBILE except aeronautical mobile	

Allocation tables

Oneweb &
SpaceX/Starlink

ITU-R Radio Regulations 2016

10.7-11.7 GHz		
Allocation to services		
Region 1	Region 2	Region 3
10.7-10.95 FIXED FIXED-SATELLITE (space-to-Earth) 5.441 (Earth-to-space) 5.484 MOBILE except aeronautical mobile	10.7-10.95 FIXED FIXED-SATELLITE (space-to-Earth) 5.441 MOBILE except aeronautical mobile	
10.95-11.2 FIXED FIXED-SATELLITE (space-to-Earth) 5.484A 5.484B (Earth-to-space) 5.484 MOBILE except aeronautical mobile	10.95-11.2 FIXED FIXED-SATELLITE (space-to-Earth) 5.484A 5.484B MOBILE except aeronautical mobile	
11.2-11.45 FIXED FIXED-SATELLITE (space-to-Earth) 5.441 (Earth-to-space) 5.484 MOBILE except aeronautical mobile	11.2-11.45 FIXED FIXED-SATELLITE (space-to-Earth) 5.441 MOBILE except aeronautical mobile	
11.45-11.7 FIXED FIXED-SATELLITE (space-to-Earth) 5.484A 5.484B (Earth-to-space) 5.484 MOBILE except aeronautical mobile	11.45-11.7 FIXED FIXED-SATELLITE (space-to-Earth) 5.484A 5.484B MOBILE except aeronautical mobile	

IMT2020 / 5G

New spectrum requested (WRC-19 cycle; AI 1.13)

- | | |
|---------------|---------------|
| - 24.25–27.50 | - 47.00–47.20 |
| - 31.80–33.40 | - 47.20–50.20 |
| - 37.00–40.50 | - 50.40–52.60 |
| - 40.50–42.50 | - 66.00–76.00 |
| - 42.50–43.50 | - 81.00–86.00 |
| - 45.50–47.00 | GHz |

IMT2020 / 5G

New spectrum requested (WRC-19 cycle; AI 1.13)

- | | |
|---------------|---------------|
| – 24.25–27.50 | – 47.00–47.20 |
| – 31.80–33.40 | – 47.20–50.20 |
| – 37.00–40.50 | – 50.40–52.60 |
| – 40.50–42.50 | – 66.00–76.00 |
| – 42.50–43.50 | – 81.00–86.00 |
| – 45.50–47.00 | GHz |

Potentially affecting RAS

IMT2020 / 5G

New spectrum requested (WRC-19 cycle; AI 1.13)

- | | |
|---------------|---------------|
| - 24.25–27.50 | - 47.00–47.20 |
| - 31.80–33.40 | - 47.20–50.20 |
| - 37.00–40.50 | - 50.40–52.60 |
| - 40.50–42.50 | - 66.00–76.00 |
| - 42.50–43.50 | - 81.00–86.00 |
| - 45.50–47.00 | GHz |

Potentially affecting RAS



Favored by Europe

5G base stations: beam-forming

28 GHz, 8x8

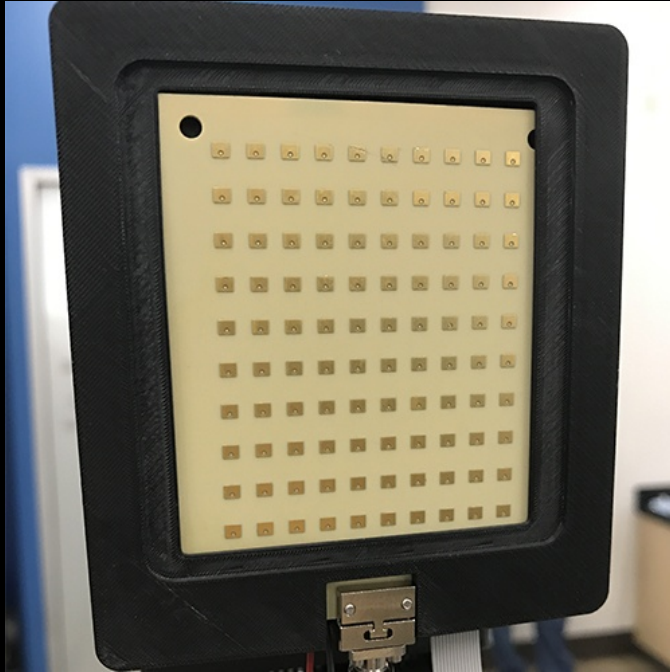
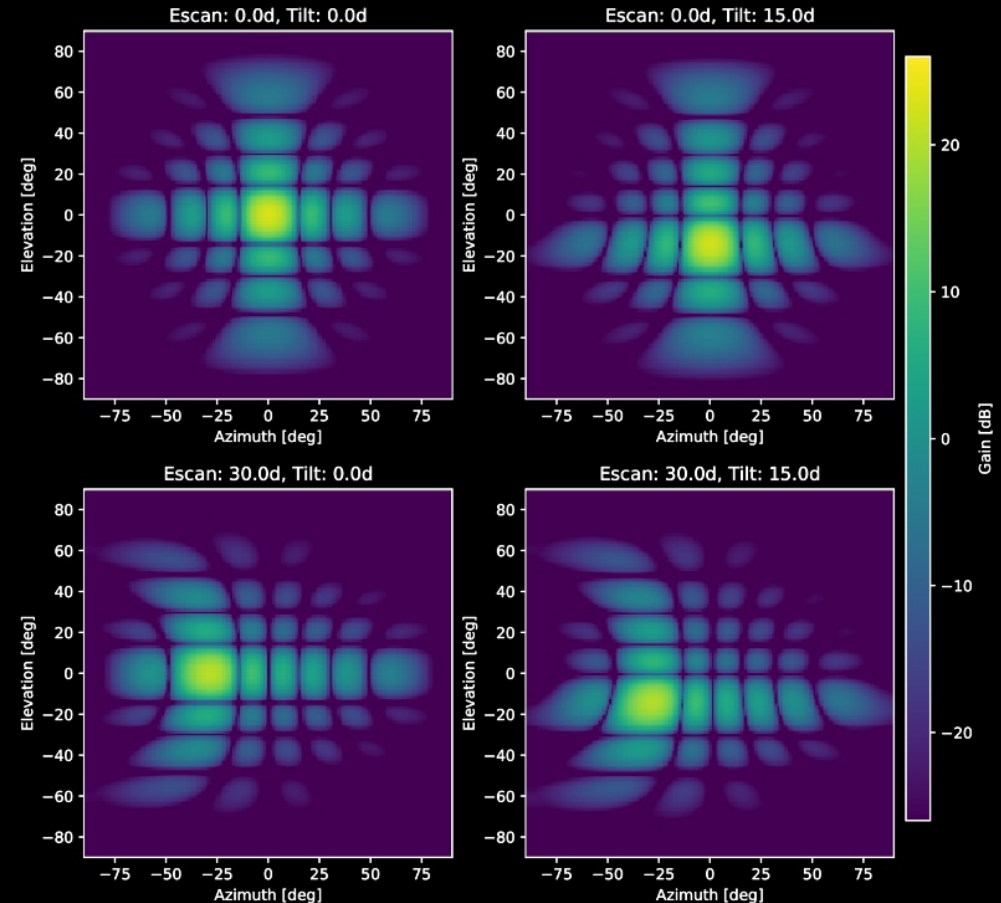
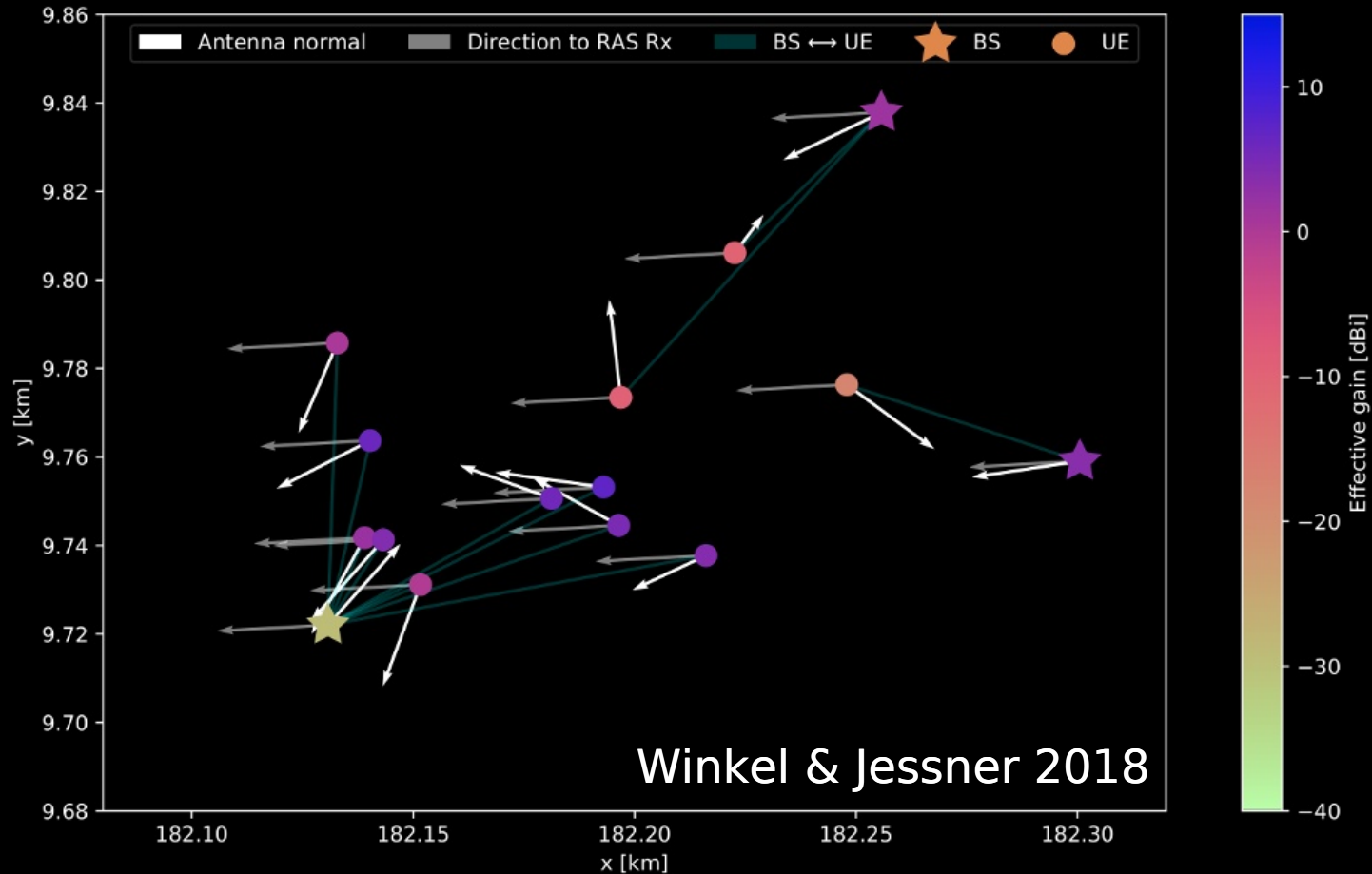


Image: UCSD



Winkel & Jessner 2018

Aggregation: simulating a network



5G: results

