“RFI mitigation OTOH”

Software RFI Mitigation for VLBI and Phased Arrays

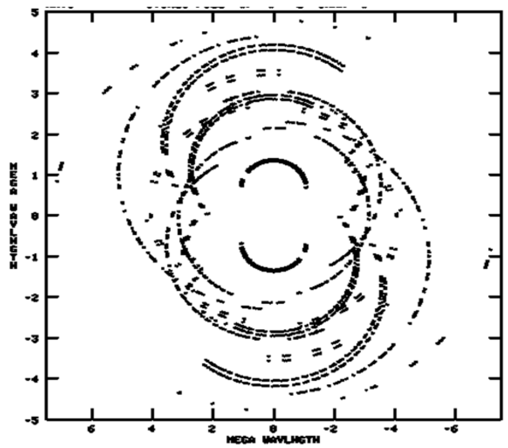
Jan Wagner
RFI Workshop, MPIfR Bonn, 8-12 April 2013

Overview

- VLBI: RFI suppression
  - Filtering out high fringe-rate components in raw visibilities
  - „RFI“ branch of DiFX Software Correlator
- Focal Plane and Phased Arrays: offline RFI subtraction
  - Make „RFI Template“ from time-averaged data, subtract
  - Trial library in DiFX Software Correlator
  - Applicable in future (GBT K-band FPA, EB L-band 7-beam, EB APERTIF, LOFAR?) – current digital backends lack computing power / architecture needed to form cross-correlations between ant. elements

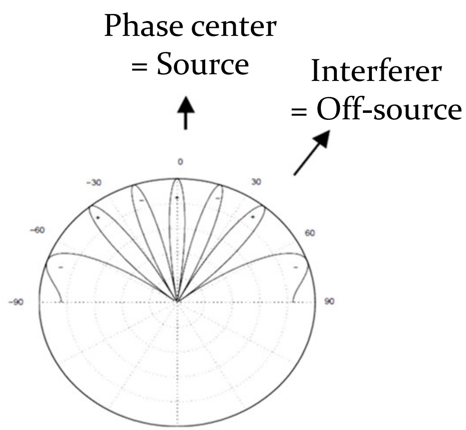
RFI Suppression for VLBI

- VLBI: spatially sample source visibility function via Earth rotation => fill sparse spatial covariance matrix of the antenna array ≈ C(t|u,v) , “gridded visibilities“



Spatial covariance matrix C(t|u,v)

- Fringe stopping in VLBI: phase center tracks source while Earth rotates – done via continual re-phasing of antennas



„Fringe stopping“ is re-phasing the two antennas of a baseline towards the source under Earth rotation. Figure: Roshi 2003

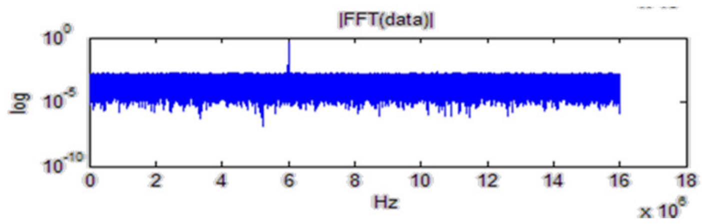
RFI Suppression for VLBI

Before fringe stopping

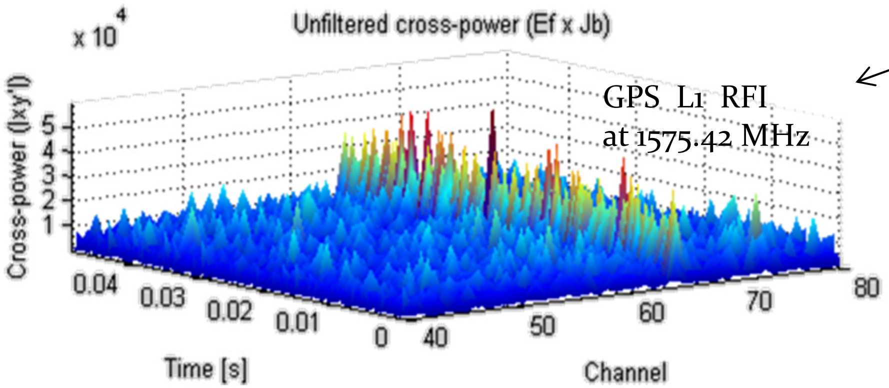
source : high fringe rate > 0 Hz

other : low fringe rate ~ 0 Hz

Locally stationary RFI, no Doppler



Synthetic Mark5B data with GPS ; Ef-Jb

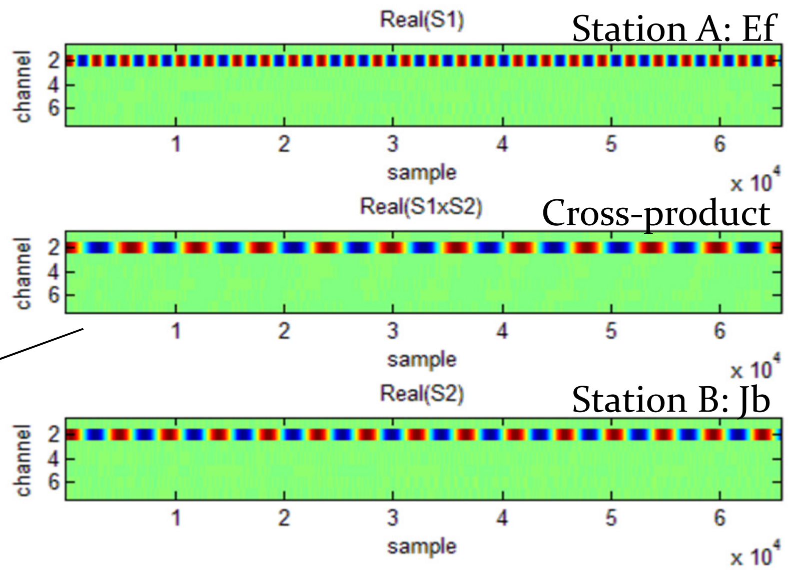


After fringe stopping

source : low fringe rate ~0 Hz

other : high fringe rate > 0 Hz

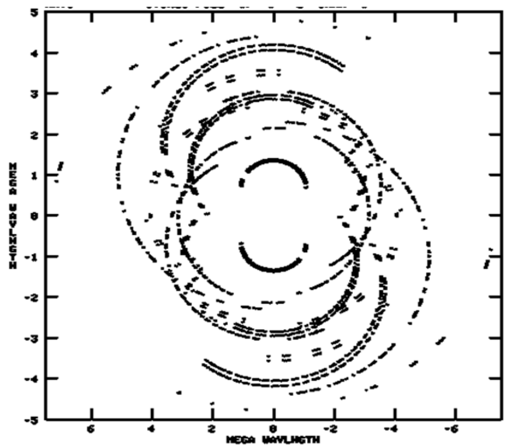
Spectrogram: fringe stopping on baseline



RFI fringe frequency ≈ baseline D, declination

RFI Suppression for VLBI

DiFX Software Correlator
- raw data is „filtered“ during AP time
- one C(*t*|u,v) sample/channel/AP gets output
  == resampling, e.g. 125 kHz down to 1 Hz

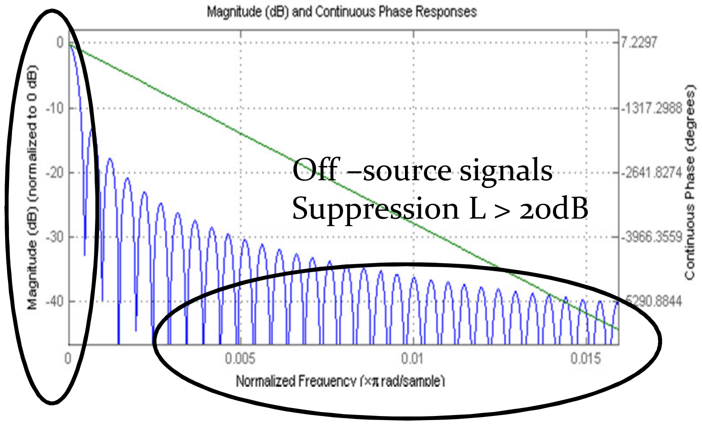


Spatial covariance matrix C(t|u,v)

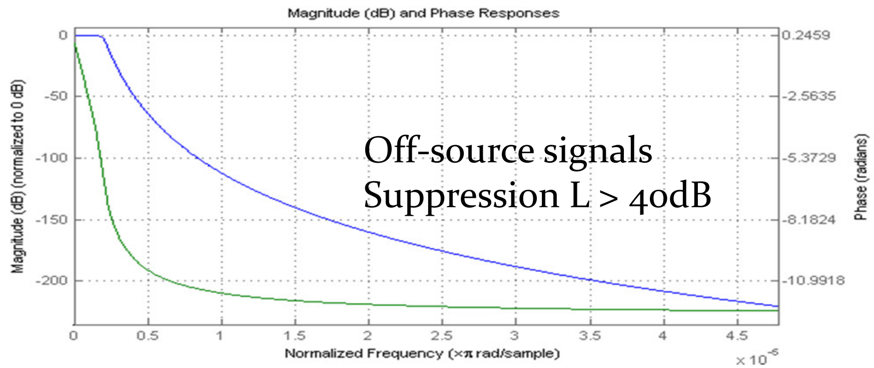
Standard DiFX : time integration in APs
RFI DiFX : low-pass (+ integration)
„fringe rate filtering“ (first suggested by Roshi et al. 2003)
~ tapering along UV track

Time integration: low-freq response

Source signal is near zero fringe freq.

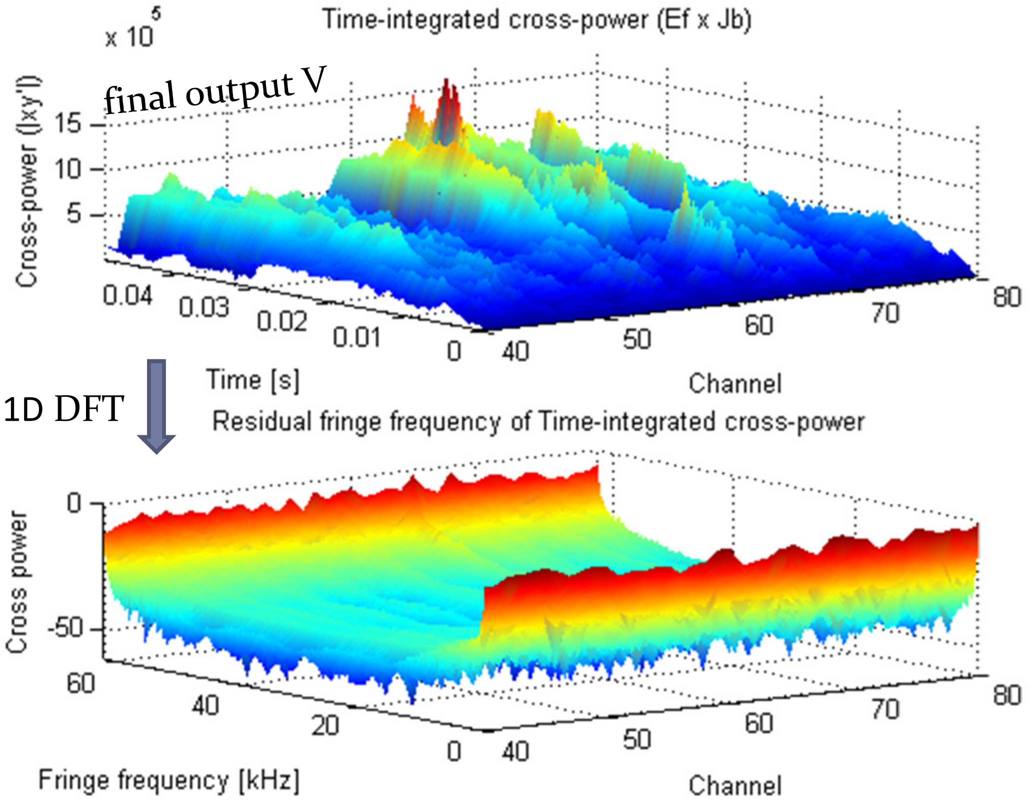


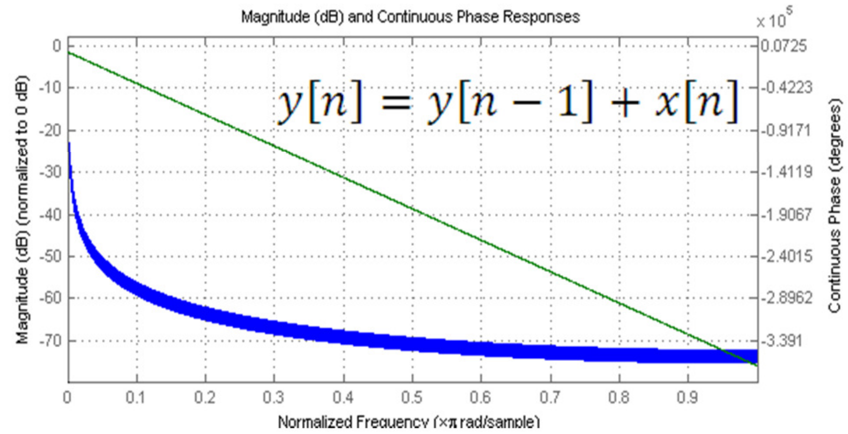
Low-pass: low-freq response, 4th order IIR

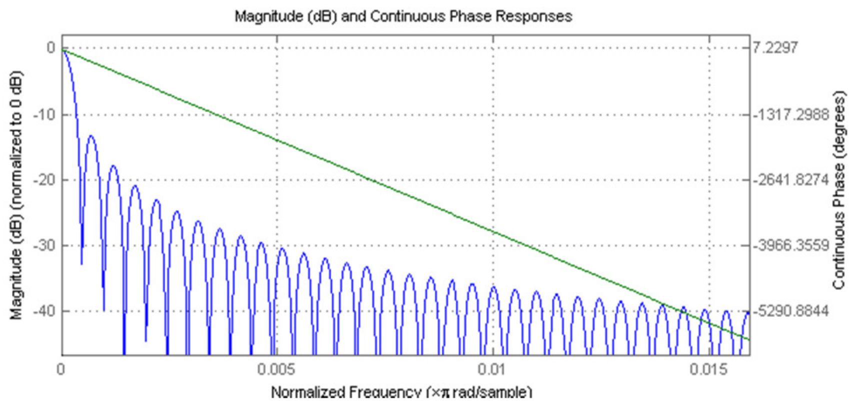


VLBI Fringe Rate Filtering: DiFX Time Integration

$$V = \langle X_A X_B^* \rangle \cong \sum_{n=0}^{Nint} x_A[n] x_B^*[n]$$ of fringe stopped data

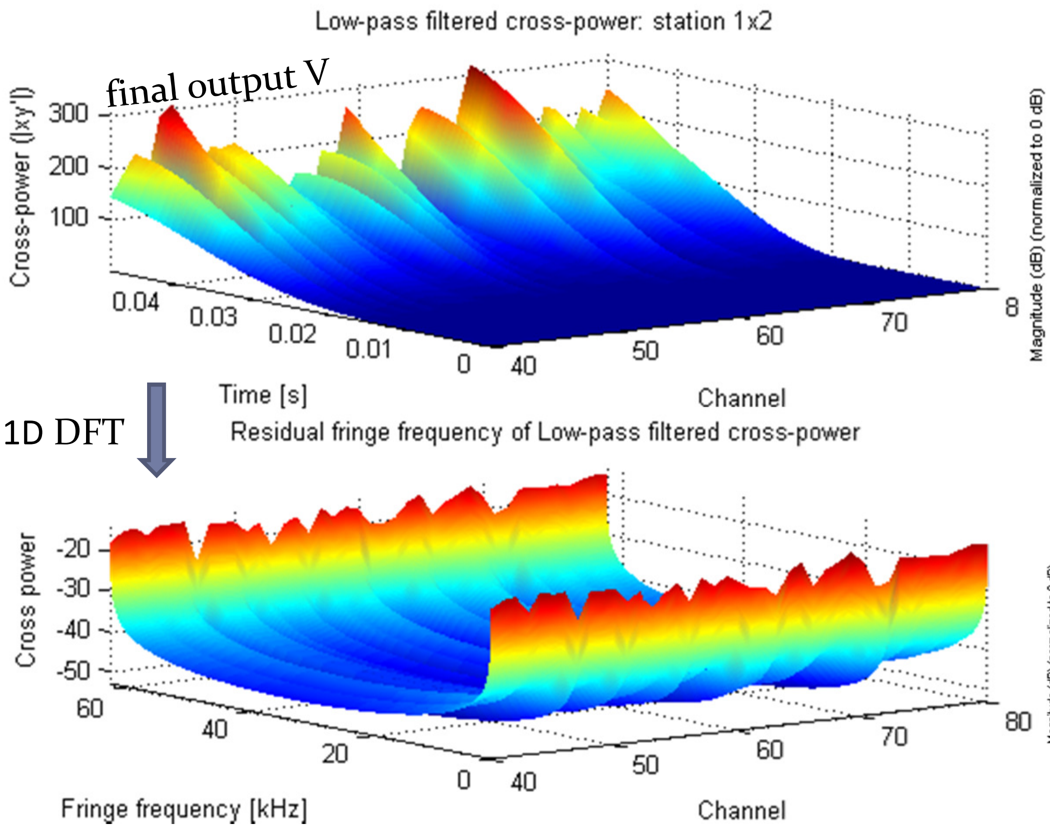


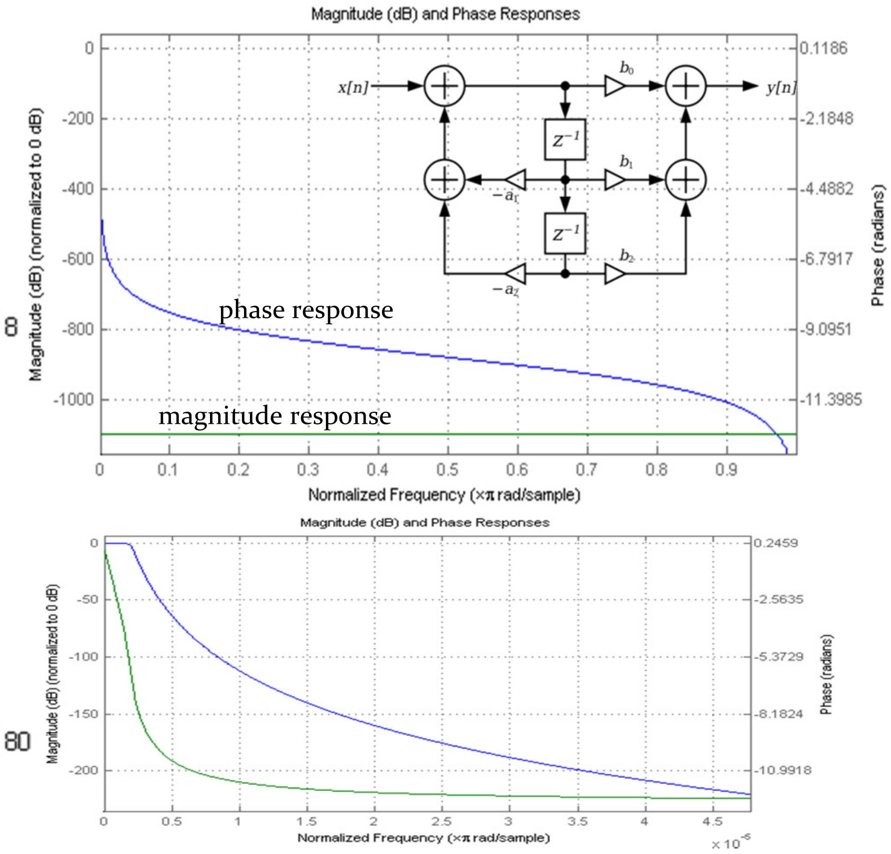




Normalized frequency response of integrator for 4096 samples length (top). Zoomed to lowest frequencies (bottom). Attenuation ~144 dB for length of several million samples. Green: phase, blue: magnitude.

VLBI Fringe Rate Filtering: DiFX Low-Pass Filters

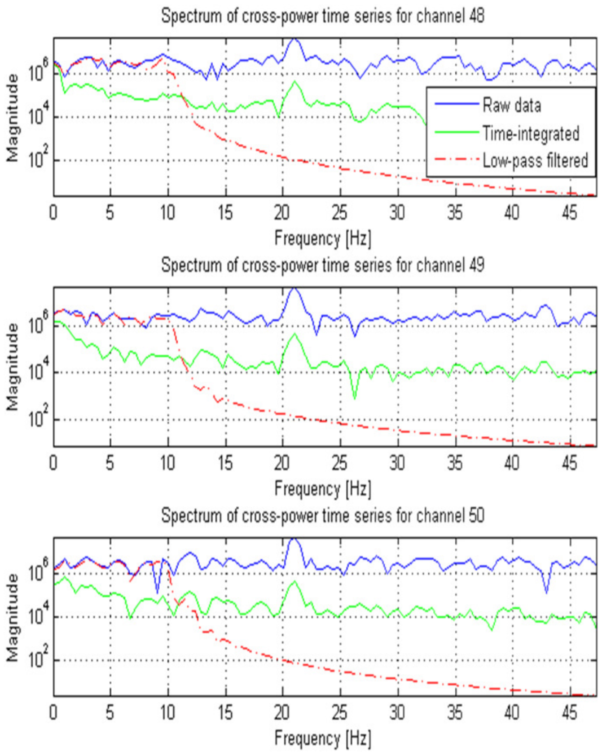




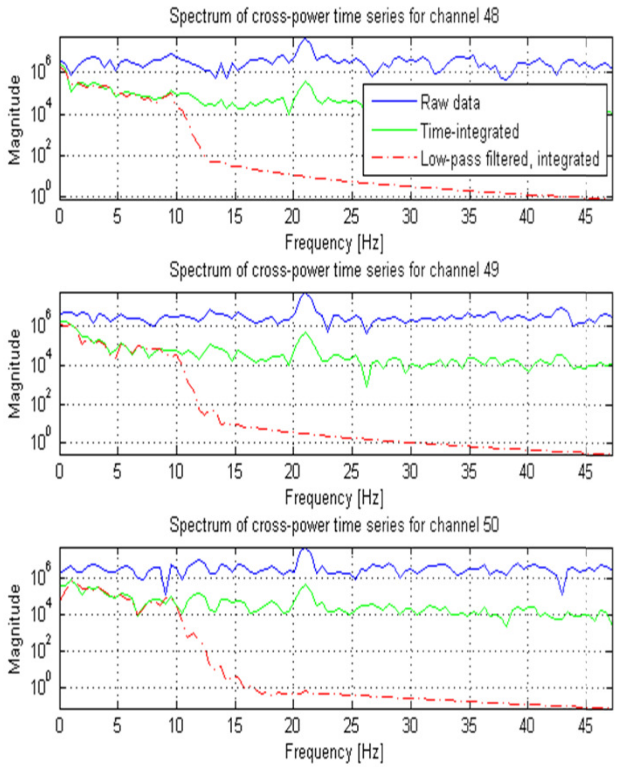
Normalized frequency response of 4th orderIIR lowpass (top). Zoomed to lowest frequencies (bottom).Green is phase and blue is magnitude response. Single precision has less atten.

VLBI Fringe Rate Filtering

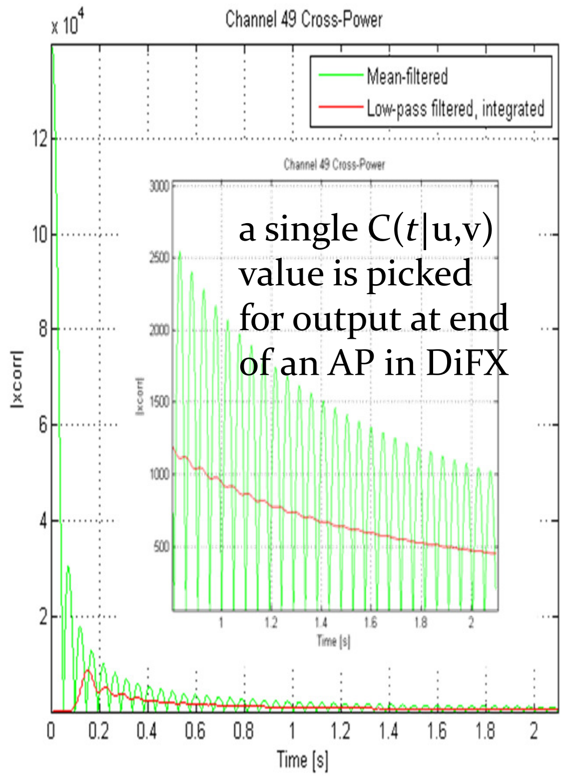
Cross-corr. with synthetic Mark5B with GPS signal, BPSK by CA sequence
Ef-WB 330km; L-band; RA=0 DEC=+45; GPS lands at fringe freq of ~22 Hz



raw → L = 0 ; integrator → L = 20 dB
12th IIR 10 Hz LP → L = 40-50 dB



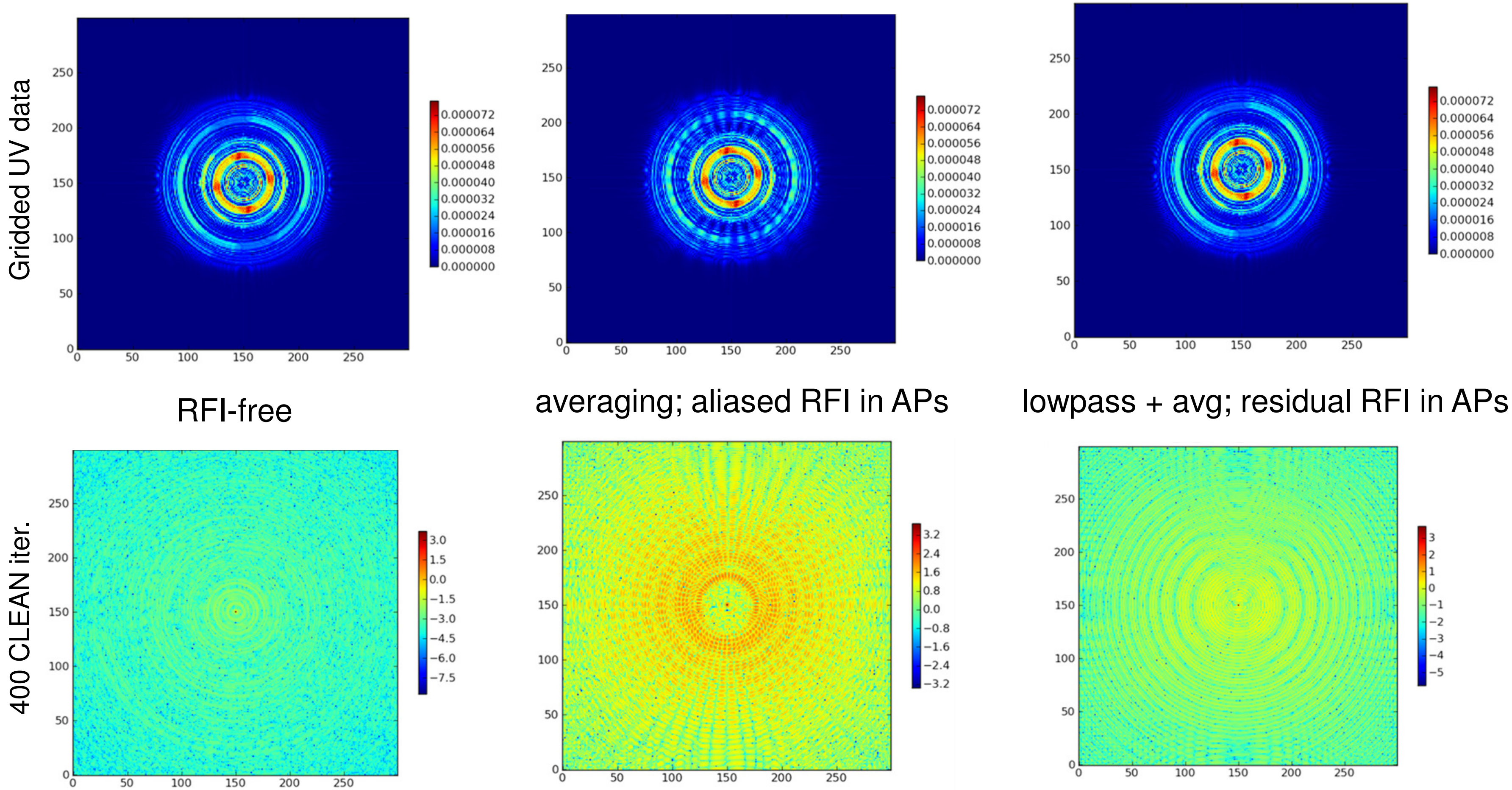
raw → L = 0 ; integrator → L = 20 dB
12th IIR 10 Hz LP + *integrator* → L = 60-70 dB



Baseline output visibility over time; inset zoomed to 0.8s-2.2s

VLBI Fringe Rate Filtering

Synthetic 4-ant. array, source near South Pole, 5 kJy, RFI on one long baseline at 1% of source flux



Focal Plane and Phased Arrays: Offline Mitigation

- Input data = covariance matrix C(t|u,v): N_{ant}x N_{ant}

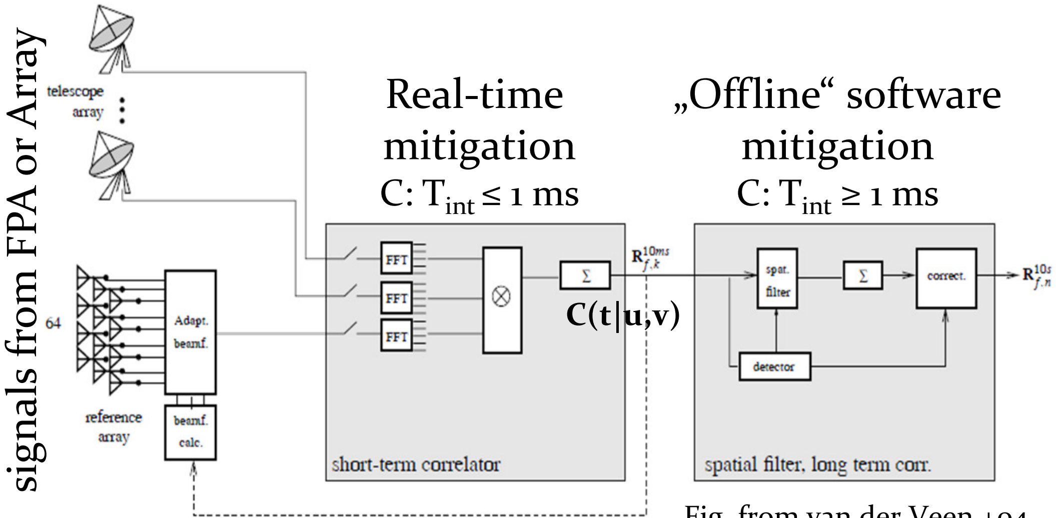
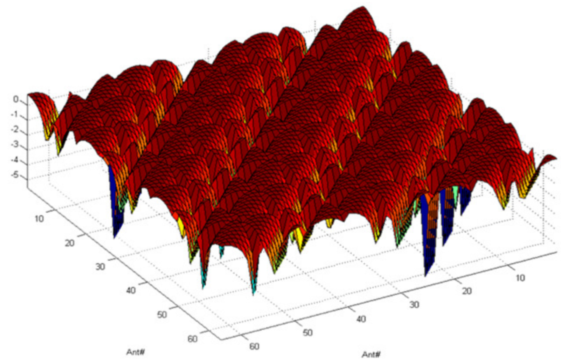
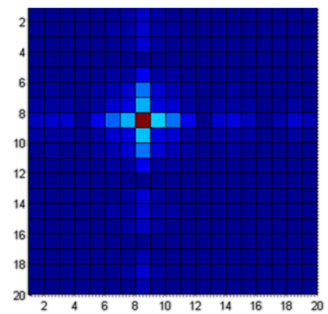


Fig. from van der Veen +04



Example covariance matrix for N_{ant}=8x8 uniform grid array
Same matrix, after imaging:

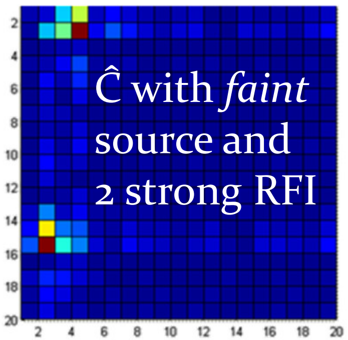


- Method 1 - RFI Subtraction : in DiFX Library
  - one or more ($N_{ref} \geq 1$)reference antennas to make „RFI Template“; must have ref. antennas part of C(t|u,v) !!
- Method 2 - RFI Nulling : also in DiFX Library
  - old but very effective when N_{ref}=0 and strong RFI

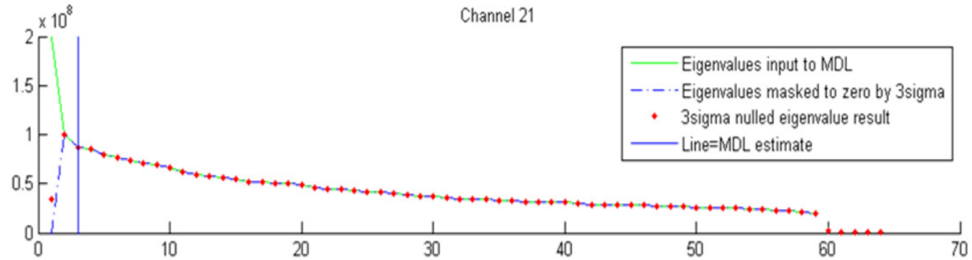
FPA RFI Nulling

- No reference antennas -- estimate #RFI from covariance $\hat{C}$: $N_{ant}xN_{ant}$
- Take matrix eigendecomposition (SVD, EVD) of $\hat{C}$ to get eigenvalues Λ:

$$\hat{C} = W \Lambda W^* = W \begin{bmatrix} \Lambda_{00} & 0 \\ 0 & \Lambda_{11} \end{bmatrix} W^*$$

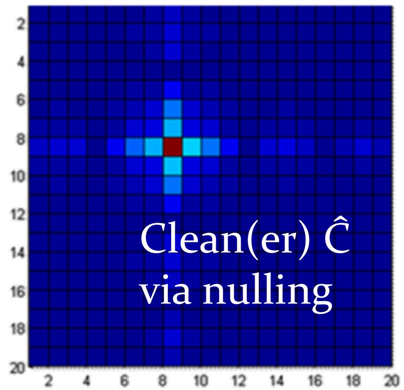


SVD of $\hat{C}$ => Λ eigenvalues

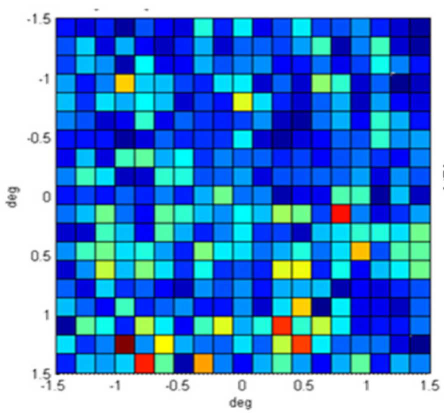


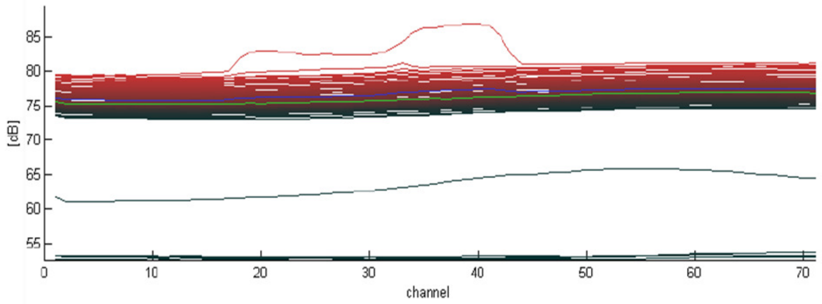
64 antennas => 64 eigenvalues

- Robustly estimate #RFI q via $q = \arg \min_{k \in [0, N_{ant}-1]} L(\Lambda | k)$
  - Minimum Description Length (Occam‘s razor)
  - 3σ threshold, MAD threshold, ...
- Replace q peak eigenvalues by non-RFI median => or by zero (=nulling)
- Old but quite effective offline mitigation!

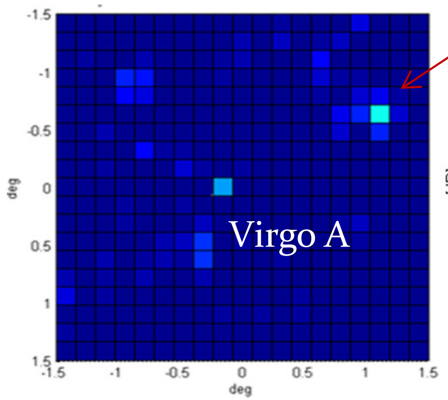


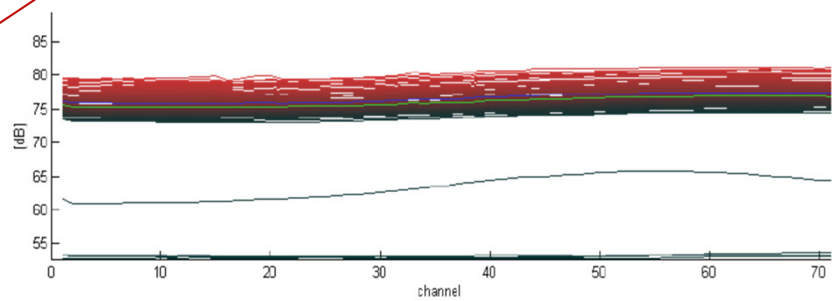
FPA RFI Nulling – APERTIF on Virgo A





weak wide-band RFI or un-ID'd other source of Markarian's Chain?





Top image: original On-Off delta covariances, rms σ=3.99. Bottom image: "nulled" covariances, rms σ=1.11. Image FOV=1.5°. WSRT main dish f/D=0.35, D=25m subtends 55°. Nulled data enhances Virgo A at ~(0°,0°). Point source at (-0.7°,1.1°) either wide-band RFI in WSRT spillover or an unidentified other source in Markarian's Chain.

Raw Ĉ :1-sec averaged On (Virgo A) and Off (sky) pointings with Wb APERTIF, provided by W.v.Cappellen

APERTIF :121 array elements, 71 channels, 1.4830 to 1.4967 GHz

Strong RFI from AfriStar
No reference antennas

Images: On-Off deltas averaged over 71 channels, before and after Nulling.

FPA RFI Subtraction

- RFI Subtraction : form template RFI footprint & subtract from data
- Need time-integrated covariance matrix $\hat{C}(f)$ incl. reference antennas:

$$\hat{C} = \begin{bmatrix} C_{rr} & C_{ra} \\ C_{ar} & C_{aa} \end{bmatrix}$$

(C_{ra} = Reference x Array, C_{aa} = Array x Array, C_{rr} = Ref x Ref)

- Closure Phase method (Kesteven, Briggs, 2000): perfect subtraction

$$\tilde{C}(i,j) = \hat{C}_{aa}(i,j) - \frac{C_{ar}(i,a_1)C_{ra}(a_2,j)C_{rr}{}^*(a_1,a_2)}{\alpha(f)+C_{rr}(a_1,a_2)C_{rr}{}^*(a_1,a_2)}$$

for $N_{ref} = 2$ and $N_{rfi} \leq 1$

- Generic method (van der Veen, 2004): good to mediocre subtraction

$$\tilde{C} = C_{aa} - \left(C_{ar}C_{rr}{}^{-1}C_{ra}\right).$$

for $N_{rfi} \leq N_{ref}$ at high INR

- Generic robust method (own): good subtraction, two alternate ways

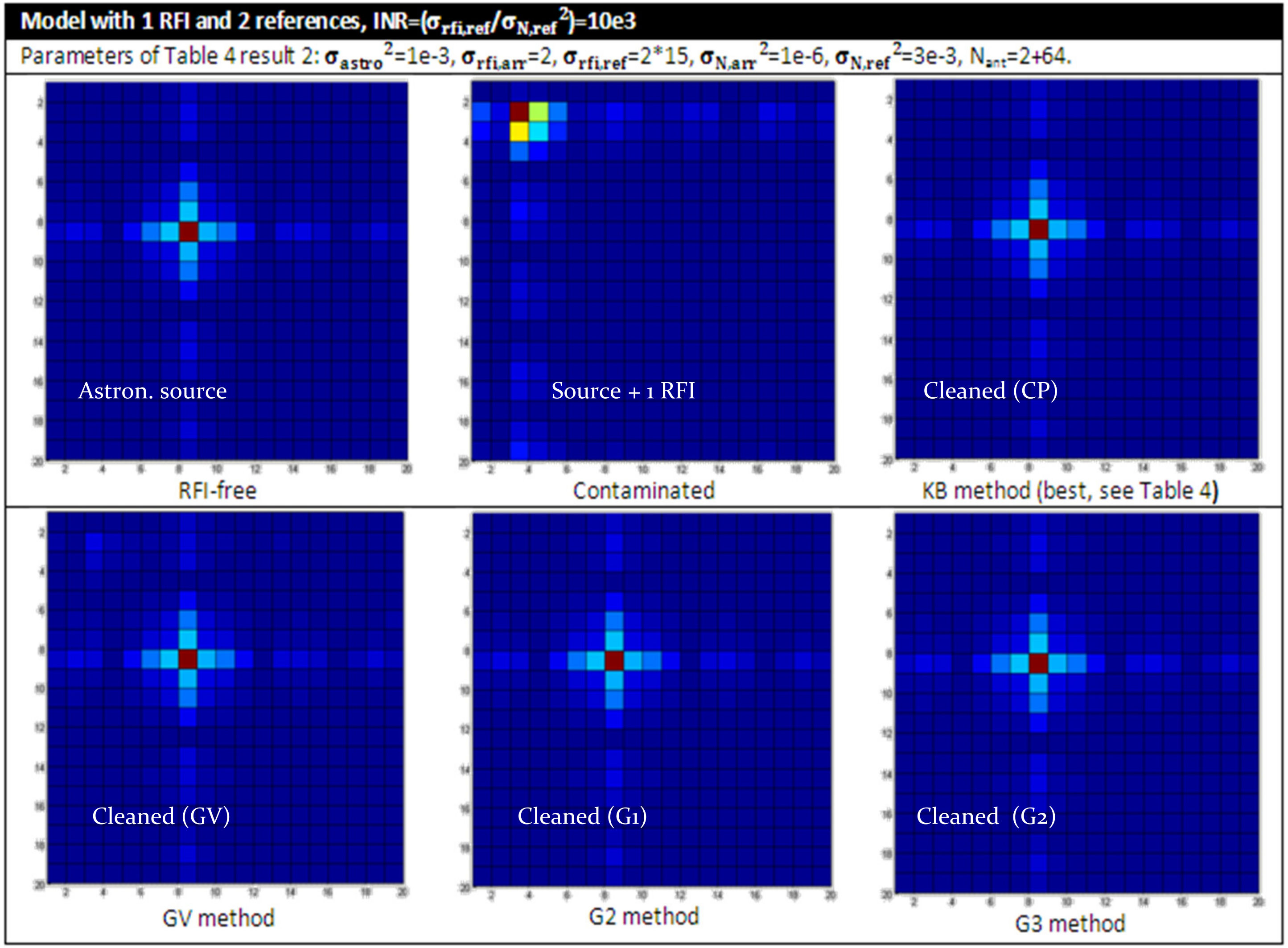
$$\tilde{C} = C_{aa} - C_{ar}\left(C_{ra}C_{aa}{}^{-1}C_{ar}\right)^{\dagger}C_{ar}$$

$$\tilde{C} = C_{aa} - \left(C_{ar}C_{ar}{}^{\dagger}\right)C_{aa}\left(C_{ar}C_{ar}{}^{\dagger}\right)^{*}$$

for $1 \leq N_{ref} \leq N_{rfi}$ down to low INR

Details at: http://www.radionet-eu.org/fp7wiki/doku.php?id=jra:albius:processing

RFI Mitigation for Phased Arrays – Synthetic Results



UV images 64-Antenna Array

1 RFI, *high* INR

Top 1: Clean
Top 2: Added RFI

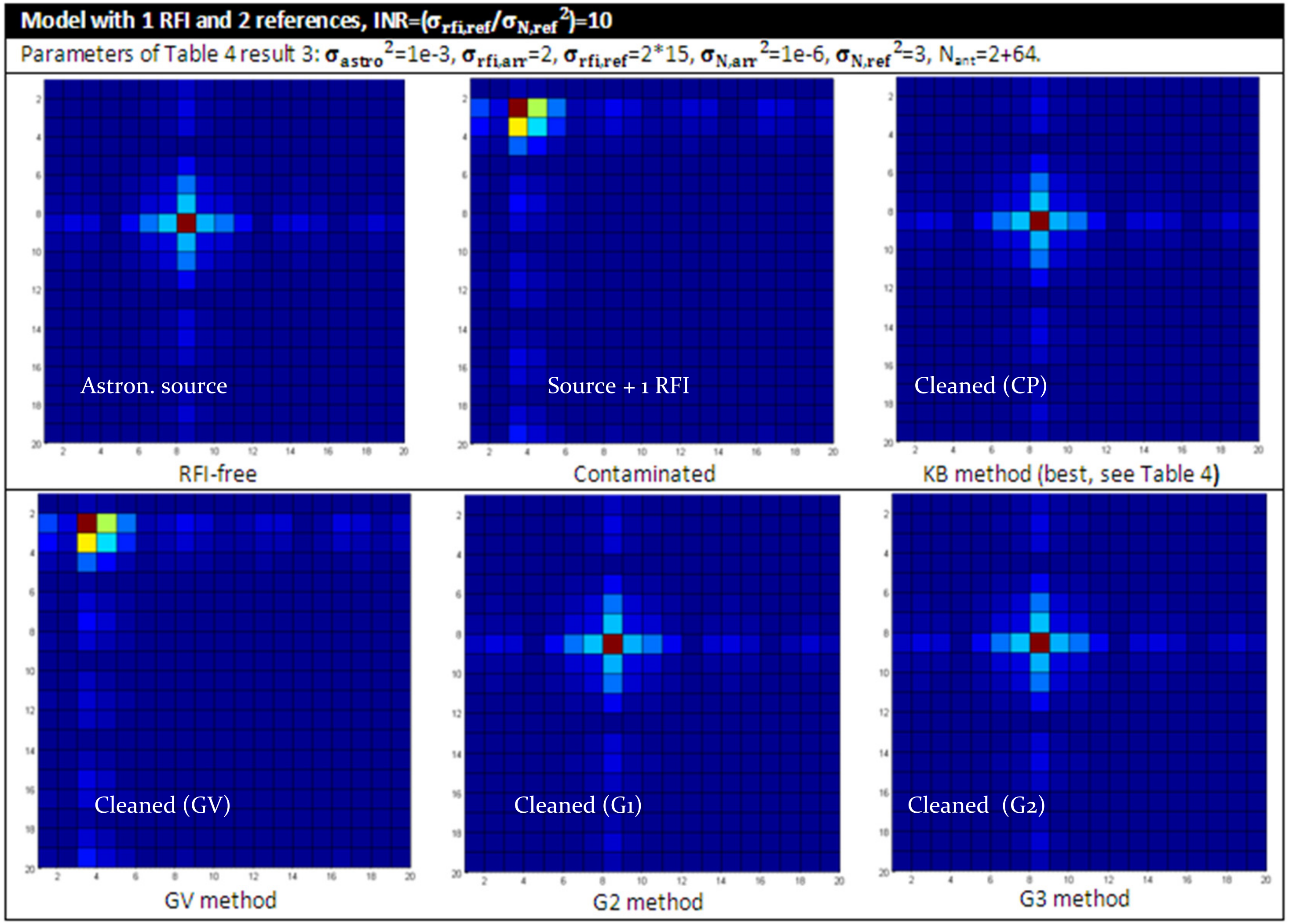
Top 3, Bottom 1-3: different cleaning methods

CP/KB = Closure phase, Kesteven

GV = Generic, Van der Veen

G1,G2 = Generic, own

RFI Mitigation for Phased Arrays – Synthetic Results



UV images 64-Antenna Array

1 RFI, *low* INR

Top 1: Clean
Top 2: Added RFI

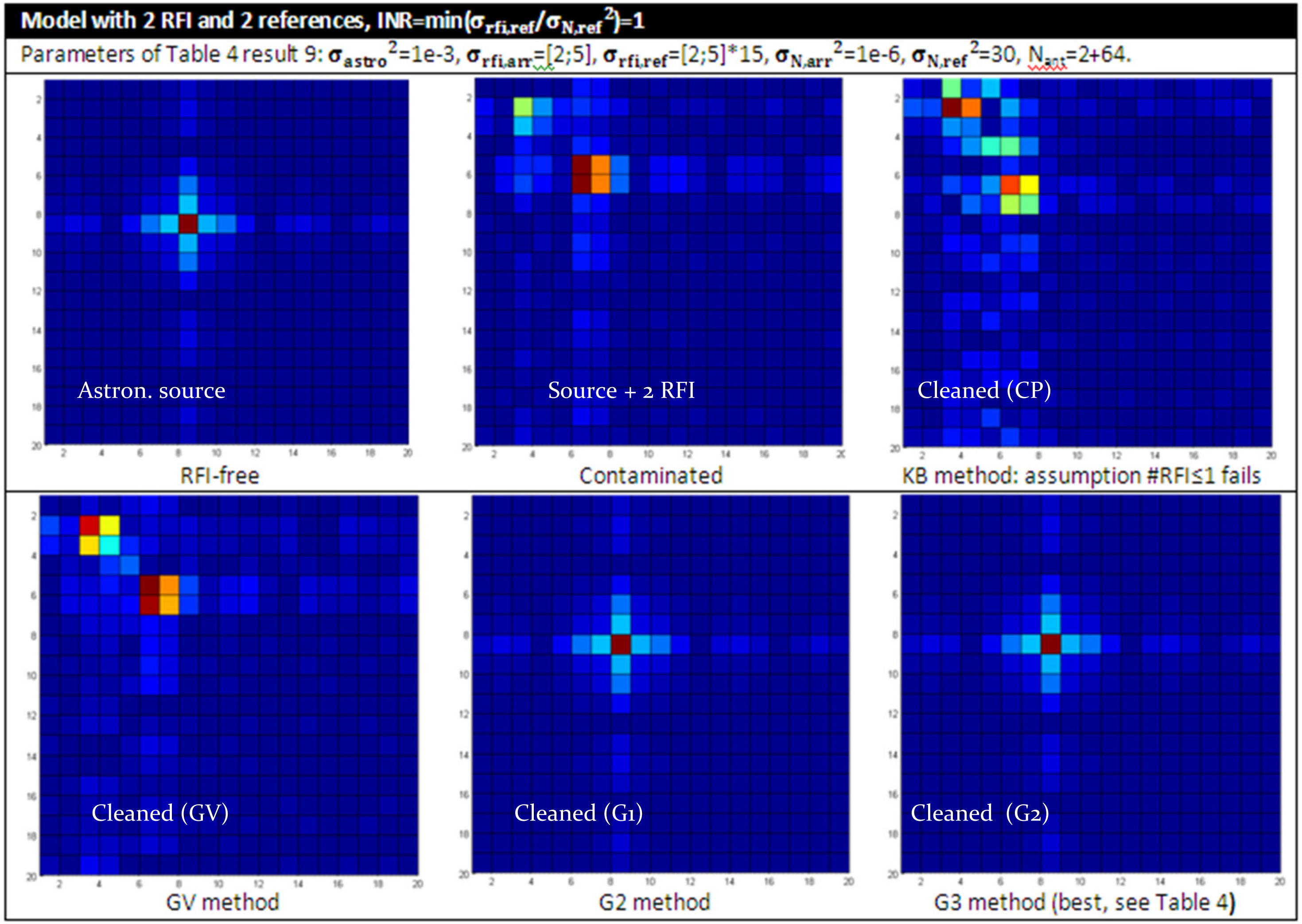
Top 3, Bottom 1-3: different cleaning methods

CP/KB = Closure phase, Kesteven

GV = Generic, Van der Veen

G1,G2 = Generic, own

RFI Mitigation for Phased Arrays – Synthetic Results



UV images 64-Antenna Array

2 RFI, *low* INR

Top 1: Clean
Top 2: Added RFI

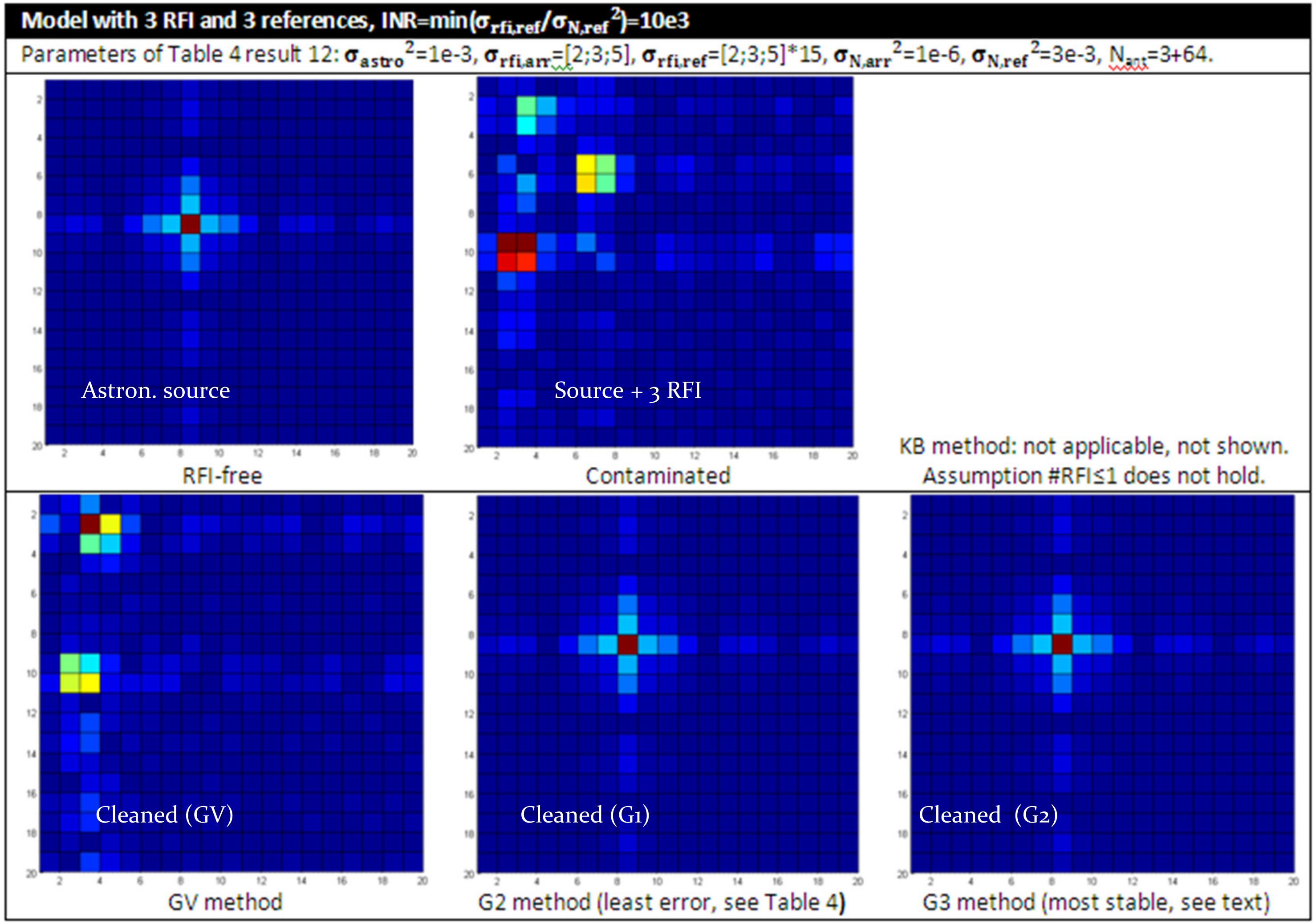
Top 3, Bottom 1-3: different cleaning methods

CP/KB = Closure phase, Kesteven

GV = Generic, Van der Veen

G1,G2 = Generic, own

RFI Mitigation for Phased Arrays – Synthetic Results



UV images 64-Antenna Array

3 RFI, high INR

Top 1: Clean
Top 2: Added RFI

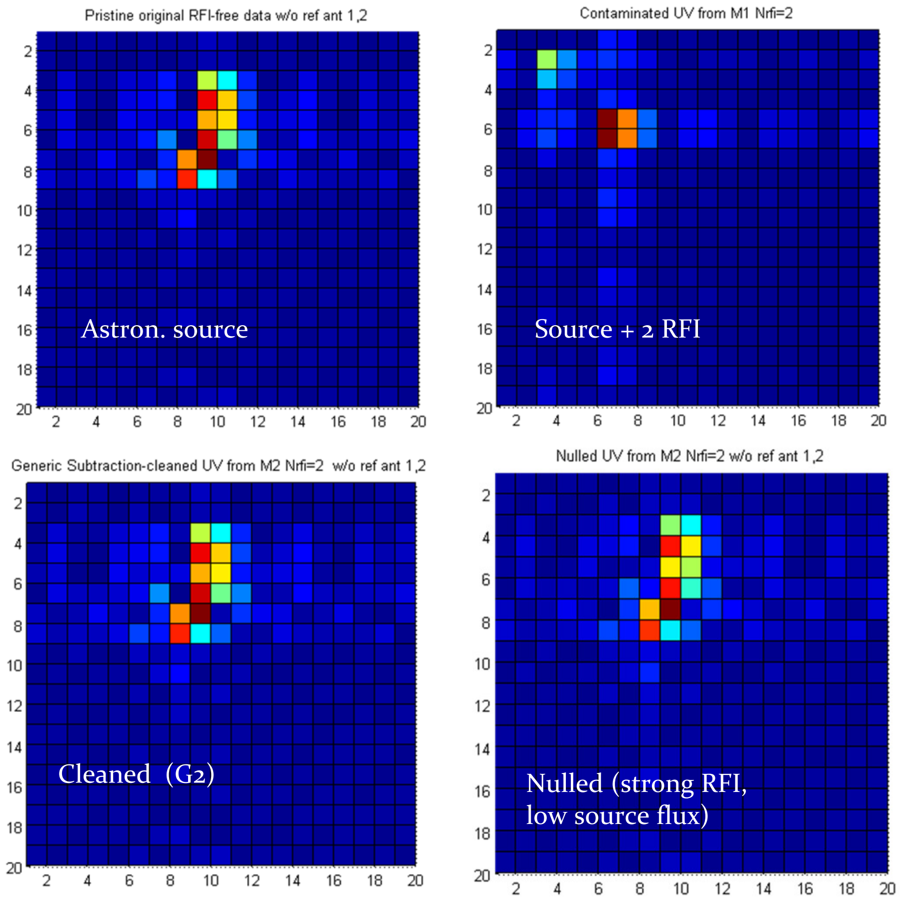
Top 3, Bottom 1-3: different cleaning methods

CP/KB = Closure phase, Kesteven

GV = Generic, Van der Veen

G1,G2 = Generic, own

RFI Mitigation for Phased Arrays – Synthetic Results



UV images 64-Antenna Array

2 RFI, high INR

Extended source

Top 1: Clean
Top 2: Added RFI

Bottom 1: Generic 2
Bottom 2: Nulled

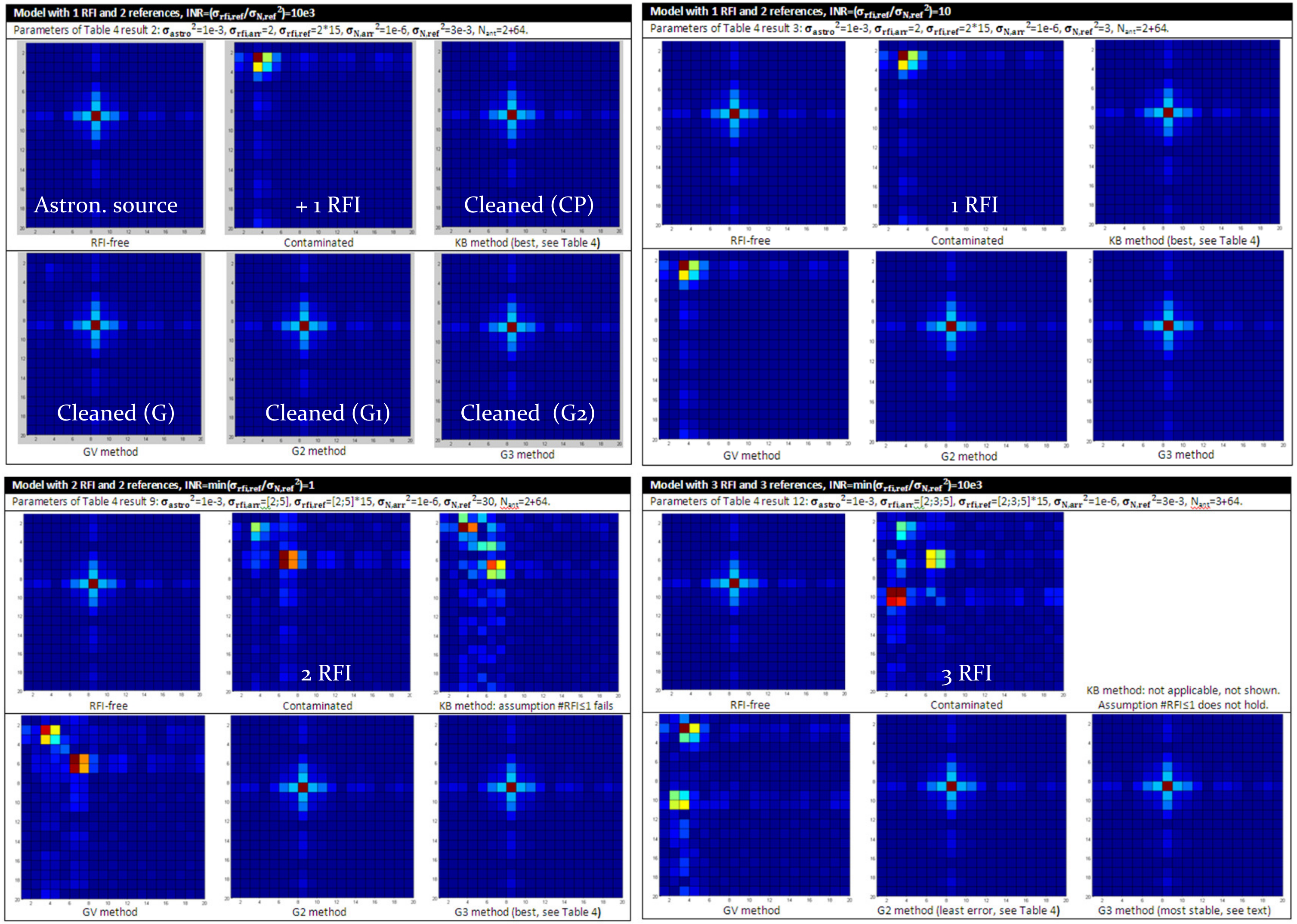
Summary

- VLBI: RFI suppression
  - In the „RFI“ branch of DiFX Software Correlator
  - In principle working, but leaves residual RFI „power“ in channels, not so interesting for spectral line VLBI!

- Focal Plane and Phased Arrays: offline RFI subtraction
  - Test library in „RFI“ branch of DiFX Software Correlator
  - Even basic „Nulling“ quite powerful
  - Reference antenna methods similarly powerful, plus carry much reduced risk of harming astronomical signal
  - But: fresher back-end hardware required to employ in real, running observations (HI surveys etc)!

Thanks!

RFI Mitigation for Phased Arrays – Synthetic Results – Overview



UV images of Covariance Data from 64-Antenna Array

Panels 1-4:

Increasing RFI, different interferer to noise ratios

Images (cw):

Top 1: Clean
Top 2: Added RFI
Top 3, Bottom 1-3: different cleaning methods

CP/KB = Closure phase, Kesteven
GV = Generic, Van der Veen
G1,G2 = Generic, own

Error Comparison

	σ_{astro}^2	$\sigma_{rfi,arr}^2$	$\sigma_{rfi,ref}^2$	$\sigma_{N,arr}^2$	$\sigma_{N,ref}^2$	N_{ant}	|ε(C)|	INR	ε(KB)	ε(GV)	ε(G2)	ε(G3)
		single RFI				**2 refs**						
1)	1e-3	2	x 15	1e-6	1e-6	2+64	1e-9	30e6	**2.2e-11**	3.8e-8	1.5e-8	5.5e-8
2)	1e-3	2	x 15	1e-6	**3e-3**	2+64	1e-9	10e3	**2.2e-11**	1.0e-4	1.8e-8	5.5e-8
3)	1e-3	2	x 15	1e-6	**3**	2+64	1e-9	10	**2.2e-11**	9.5e-2	2.0e-8	5.5e-8
4)	1e-3	2	x 15	1e-6	3	2+64	**1e-6**	10	**4.1e-11**	9.5e-2	1.5e-1	2.1e-5
		dual RFI				**≥2 refs**						
5)	1e-3	[2 ; 5]	x 15	1e-6	1e-6	2+64	1e-9	30e6	7.0	2.8e-7	1.5e-7	**7.6e-8**
6)	1e-3	[2 ; 5]	x 15	1e-6	**3**	2+64	1e-9	10	7.0	0.72	1.5e-7	**7.6e-8**
7)	1e-3	[2 ; 5]	x 15	1e-6	3	**3+64**	1e-9	10	---	0.41	1.5e-7	**7.6e-8**
8)	1e-3	[2 ; 5]	x 15	1e-6	3	**4+64**	1e-9	10	---	0.19	1.5e-7	**7.6e-8**
9)	1e-3	[2 ; 5]	x 15	1e-6	**30**	**2+64**	1e-9	1	7.0	3.44	1.5e-7	**7.6e-8**
10)	1e-3	[2 ; 5]	**x 3e-3**	1e-6	30	2+64	1e-9	1e-4	7.0	6.99	1.5e-7	**7.6e-8**
		triple RFI				**≥3 refs**						
11)	1e-3	[2 ; 3 ; 5]	x 15	1e-6	1e-6	3+64	1e-9	30e6	---	2.8e-5	**2.2e-6**	4.1e-6
12)	1e-3	[2 ; 3 ; 5]	x 15	1e-6	**3e-3**	3+64	1e-9	10e3	---	8.4e-2	**2.2e-6**	4.1e-6
13)	1e-3	[2 ; 3 ; 5]	x 15	1e-6	**3**	3+64	1e-9	10	---	6.02	**2.2e-6**	4.1e-6
14)	1e-3	[2 ; 3 ; 5]	x 15	1e-6	**1e-6**	**4+64**	1e-9	30e3	---	4.6e-6	**8.0e-7**	4.1e-6
15)	1e-3	[2 ; 3 ; 5]	x 15	1e-6	1e-6	**5+64**	1e-9	30e3	---	1.3e-6	**5.5e-7**	4.1e-6

Table 4 – Performance comparison of all four subtraction methods for the same 64-element antenna array. Rows show signal power level inputs and the emulated covariance estimation error, number of antennas (reference + array), the derived smallest interference to noise ratio (INR) in the references and the resulting max. abs. errors ε of each method relative to the RFI-free model. Not all cases are physical. Changed input values and lowest errors are indicated by bold type.

DiFX / RFI Library Performance

Single-core throughput on Xeon E5430 ; 64 antennas
One channel == one covariance matrix

$ # Armadillo with ATLAS, Beamformer compiled '-g –O3 –Wall -DUSE_SINGLE_PRECISION=1' $ numactl –physcpubind=0 ./benchmark	
Integrate 64-elem vector into Covariance	156000 channels/sec (better use FPGA or GPU!)
Decomposition -> recomposition (average)	270 channels/sec
SVD -> RFI detect -> null -> recomposition	190 channels/sec
EVD -> RFI detect -> null -> recomposition	310 channels/sec
1-RFI/ch, 2-reference Template subtraction	8700 channels/sec
2-RFI/ch, 2-reference Template subtraction	21900 channels/sec
64-beam classical beamformer	5850 channels/sec
64-beam MVDR (Cox b=1.0)	390 channels/sec
64-beam RB-MVDR (Cox b=1.0+1e-4)	360 channels/sec

DiFX Configuration File Changes: *.v2d

Example vex2difx *.v2d descriptor file with RFI-related parameter highlighted

```
#  Template v2d file for DiFX correlation of W3OH
vex = w3oh.vex.clocks
antennas = EF, JB, WB
singleScan = True

# The nChan should never be less than 128.
# For numbers of channels < 128, set specAvg so nChan/specAvg
# gives the desired number of channels
SETUP default
{
  tInt = 2.048
  nChan = 2048
  nFFTChan = 2048
  xmacLength = 2048 # to prevent FFT division into 16 x 128-channel pieces
  doPolar = True

  # enable RFI filters (if unspecified, defaults to False/off)
  doRFI = True

  # RFI filtering is only applied inside sub-integration time intervals.
  # Final summation of the sub-integration outputs does not use a filter.
  # To reduce aliasing you may increase the sub-integration time (at cost of RAM).
  # Subint of 0.512s will produce 4 subints for the full 2.0s Tint.
  subintNS = 512000000 # optional
}

# This, along with SETUP default above, should always be done
RULE default
{
  setup = default
}
```

DiFX Configuration File Changes: *.input

Extra fields in RFI DiFX *.input file compared to trunk DiFX

```
# COMMON SETTINGS ##!
CALC FILENAME:       /Exps/w3oh/w3oh_01.calc
CORE CONF FILENAME:  /Exps/w3oh/w3oh_01.threads
EXECUTE TIME (SEC):  120
START MJD:           55638
START SECONDS:       59651
ACTIVE DATASTREAMS:  3
ACTIVE BASELINES:    3
VIS BUFFER LENGTH:   80
OUTPUT FORMAT:       SWIN
OUTPUT FILENAME:     /Exps/w3oh/w3oh_01.difx
RFI FILT TYPE:       CHAIN
RFI FILT COEFFS:     /Exps/w3oh/w3oh_01.coeff

# CONFIGURATIONS ###!
NUM CONFIGURATIONS:  1
CONFIG NAME:         W3OH_default
INT TIME (SEC):      2.048000
SUBINT NANOSECONDS:  512000000
...
```

DiFX Configuration File Additon: *.coeff

```
# ----- Example filter configuration file
# Filter types: 0 = Integrator, 1 = Integer Decimator, 2 = IIR biquad, 3 = FIR,
# 4 = Digital State Variable Filter (DSVF), 5 = Moving Average

5 # Number of filters in series, with type and settings in the order below:

# ----- Integrator
0 # type: 0=integrator, has no coefficients

# ----- Integer factor Decimator
1 # type: 1=decimator
3 # decimation ratio

# ----- Biquad IIR Filter
2 # type: 2=IIR-SOS/biquad
4 # filter order, order 4 requires two 2nd order section
9.99831172521226110e-006 # input prescaling gain
# filter coefficients, b0(1) b1(1) b2(1) a0(1) a1(1) a2(1); etc
1.0 -1.99999451637268070 1.0 1.0 -1.99986529350280760 0.99986535310745239
1.0 -1.99999928474426270 1.0 1.0 -1.99989640712738040 0.99989646673202515

# ---- Digital state variable filter
4    # type: 4=DSVF
1.0  # input prescaling gain
0.000773 # tuning f = 2*sin(pi*(1024*(1/0.52))/16e6)
0.5      # quality q = 1/Q=1/2.0
# ---- Moving average filter
5       # type: 5=MAvg
16      # length L of window
0.0625  # input prescaling gain, thus if gain=1/L then output is a moving average
```