Resource Management with Real-Time Complexity Monitoring inSoftware-Defined Radios

Joseph D. Gaeddert

Virginia Polytechnic Institute & State University

December 2, 2010

Introduction

Motivation

Current research platform

target deployment platform

Problem:

- Significant gap between HDR and SDR (size, weight, power)
- Flexibility of SDR *should* be able to get some of this back

Introduction

Motivation

Current research platform

target deployment platform

Problem:

- Significant gap between HDR and SDR (size, weight, power)
- Flexibility of SDR *should* be able to get some of this back

Goals:

- Reduce transceiver resources (required computation)
- Maintain service quality

Introduction

Organization

In this presentation:

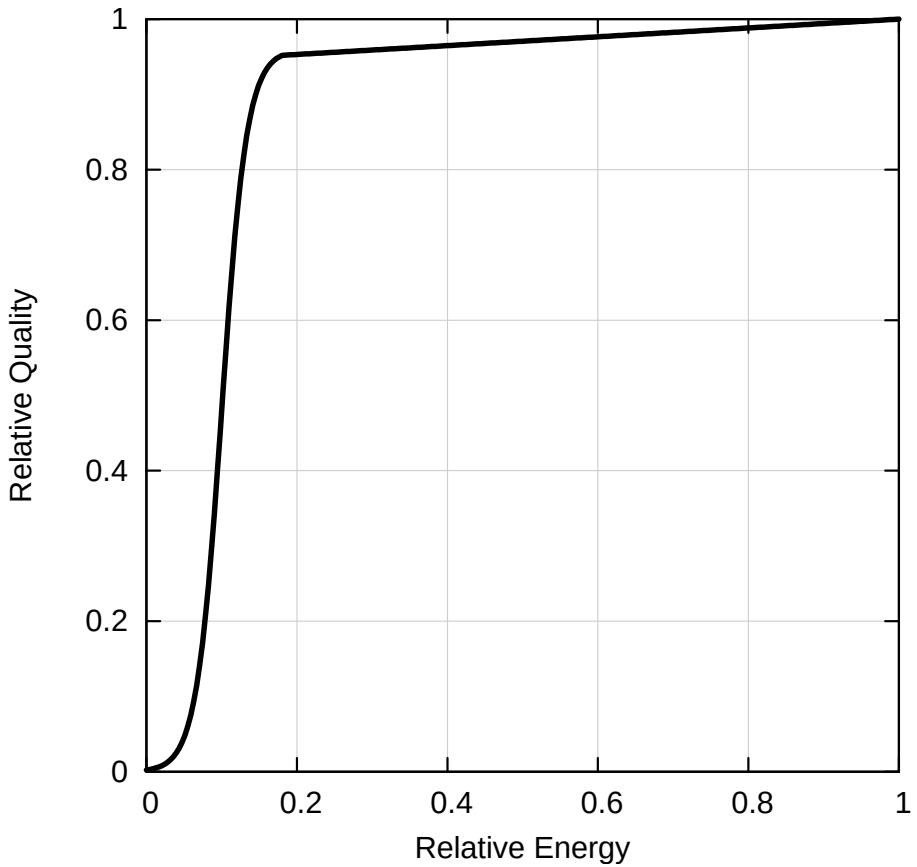
- Discuss the applicability to energy-quality scalable algorithms for SDR
- Provide a simple metric for resource efficiency in SDR
- Provide a simple model for monitoring resource consumption
- Validate simulation with over-the-air experimentation
  - adaptive modulation/coding schemes
  - online resource monitoring
- Results and conclusions

Energy-Quality Scalability

Concept

Strong non-linear relationship between energy consumed and resulting quality

- Steep initial grade
- Flattens off quickly
- Majority of quality is achieved with minimal initial energy



Energy-Quality Scalability

Applicability to SDR

Many resources involved in SDR (power, energy, spectrum, memory,. . .)

TX : power amplifier

RX : baseband processor (memory, computational bandwidth)

Energy-Quality Scalability

Applicability to SDR

Many resources involved in SDR (power, energy, spectrum, memory,. . .)

TX : power amplifier

RX : baseband processor (memory, computational bandwidth)

Resource consumption has a non-linear relationship with quality:

- processor power consumption $P_c \propto f^2$
- channel capacity: $C = W \log_2 \left(1 + \frac{\bar{P} L_P}{W N_0}\right)$

Processing Channel Efficiency

Energy Model

Define processing channel efficiency metric [b/J]

$$\eta_{\mathcal{K}} = \frac{P_c}{C} = \frac{P_c(W)}{W \log_2\left(1 + \frac{\bar{P} L_P}{W N_0}\right)}$$

P_c : consumed processor power [W]

C : channel capacity [b/s]

W : bandwidth [Hz]

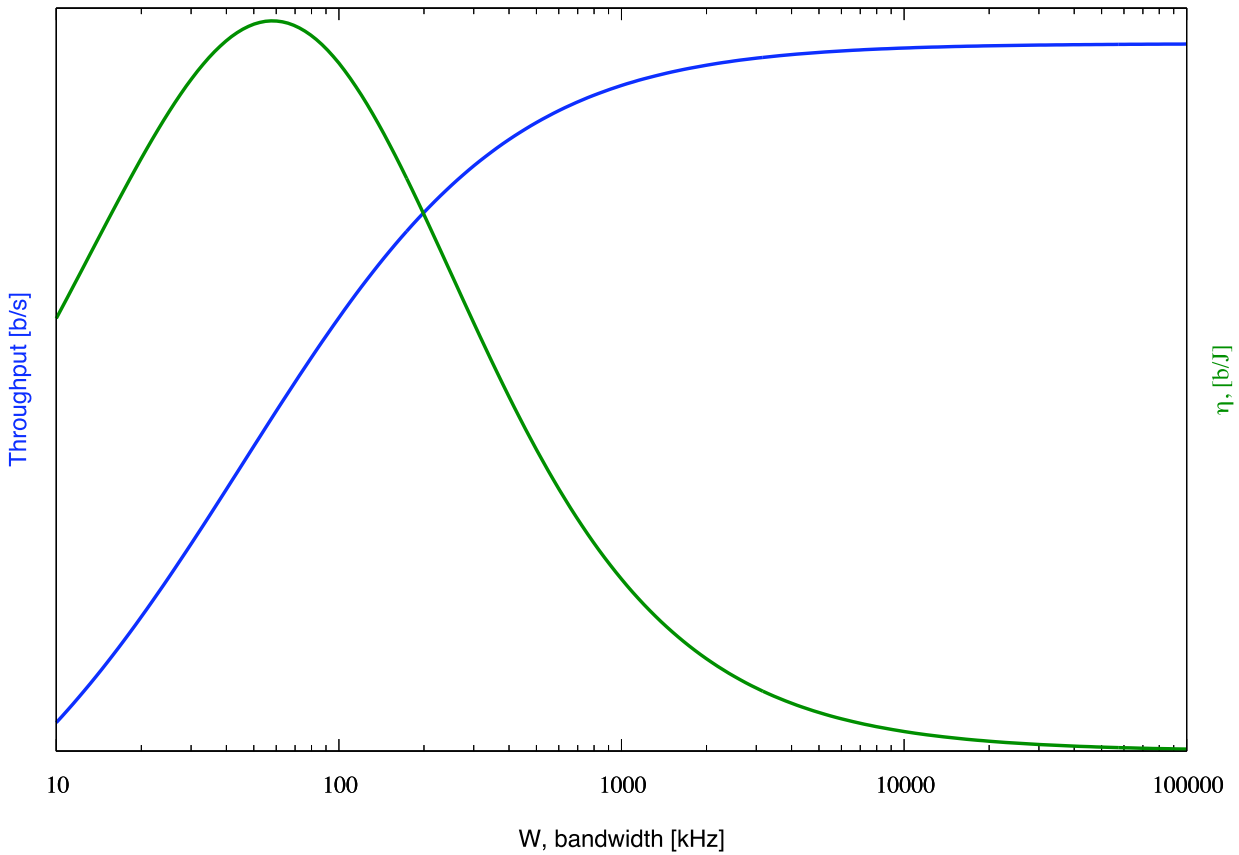
$\bar{P}$: average transmit power [W]

L_P : path loss

N_0 : noise power spectral density [W/Hz]

Channel Energy Efficiency

throughput [b/s] and energy efficiency [b/J] vs. Bandwidth



Processing Channel Efficiency

Extended model

Extend processing channel efficiency model to encorporate processor complexity

$$\bar{\mathcal{K}}(\gamma) = \frac{F_p r_p}{\eta W \left(1 - P_e(\gamma)\right)}$$

F_p : processor master clock frequency [cycles/s]

r_p : processor usage [percentage of peak load]

η : spectral efficiency [b/s/Hz]

W : occupied bandwidth [Hz]

P_e : packet error rate

γ : instantaneous packet SNR

Processing Channel Efficiency

Extended model

- average # of clock cycles to properly receive and decode a single data bit
- many of these parameters are linked (e.g. occupied bandwidth to CPU usage)
- depends upon channel conditions (SNR, fading, multipath, etc.)
- related to our previous analysis of efficiency, bits/Joule

Receiver Complexity Monitoring

For a reconfigurable receiver platform

- performance is limited by hardware restrictions
- processing complexity limits available throughput
- possible solution: source code optimization

Receiver Complexity Monitoring

For a reconfigurable receiver platform

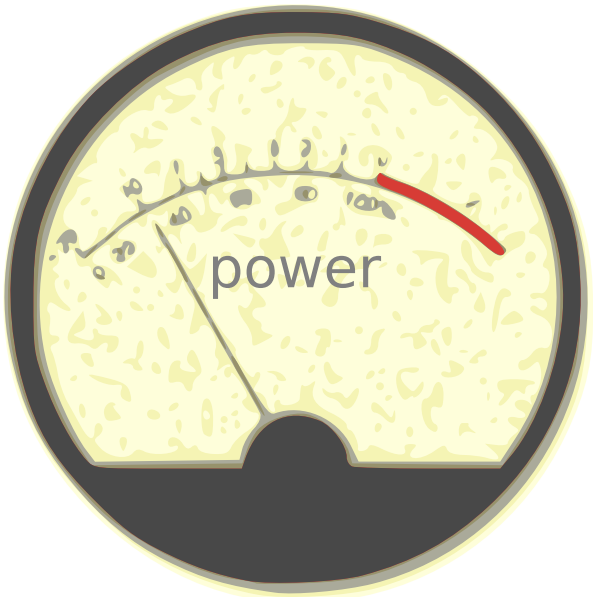
- performance is limited by hardware restrictions
- processing complexity limits available throughput
- possible solution: source code optimization

What if code optimization isn't good enough?

- change receiver structure
- receiver complexity depends significantly on algorithms
- monitor channel conditions and adjust receiver appropriately

Monitoring Real-Time Energy Consumption

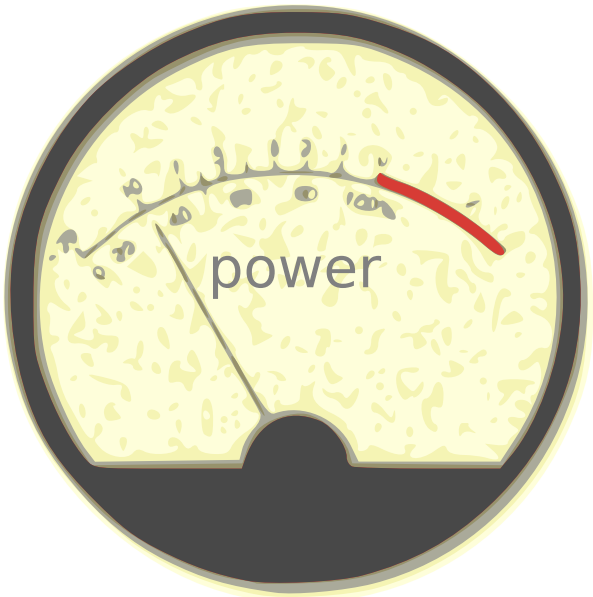
Propose: allow the radio to monitor its own power consumption and processor usage



Monitoring Real-Time Energy Consumption

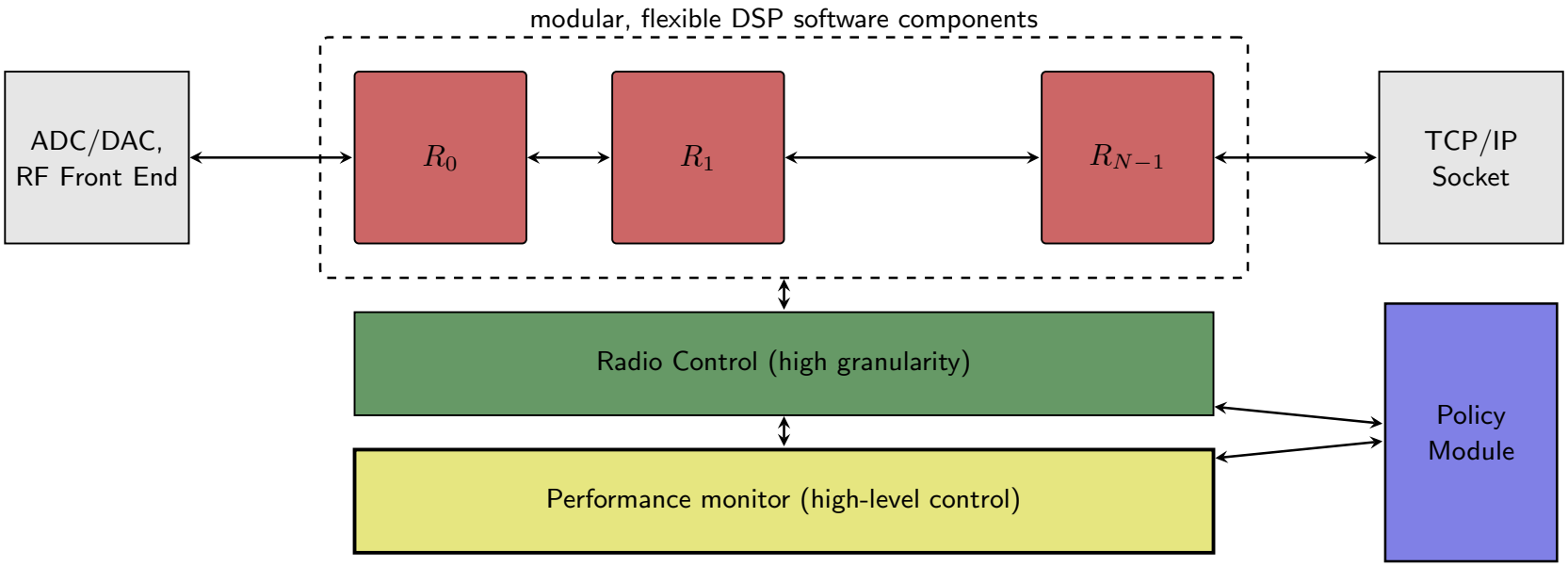
Propose: allow the radio to monitor its own power consumption and processor usage

- Performance is not a fixed value
- Data quality depends significantly on channel conditions
- Resource consumption depends highly on the same



Real-time Complexity Monitoring

Framework



Real-time Complexity Monitoring

Concept

Once code has been optimized, characterize the receiver *online*

- measure statistical complexity, $\mathcal{K}$, for different configurations
- e.g. `getrusage()` interface
- normalize complexity to throughput: $\bar{\mathcal{K}}(\gamma) = \mathcal{K}/\zeta(\gamma)$
- computational efficiency units: [clock cycles / bit]
- specific to each hardware platform
- map receiver configuration for different wireless environments

Experimentation

Adaptive Modulation/Error-Correction in Slowly-Varying Fading Channels

Adaptive radios

- switch modulation and coding schemes to maintain QoS
- performance curves sensitive to channel conditions
- complexity of receiver sensitive to modulation/coding schemes, frame length, etc.

Over-the-air Experimentation

Adaptive Modulation/Error-Correction in a Wireless Laboratory Environment

Previous results:

- Complexity measurements taken offline
- Adapted modulation/forward error-correction to Rayleigh channel
- Significant computational savings (5 fold)
- Marginal throughput degradation

Over-the-air Experimentation

Adaptive Modulation/Error-Correction in a Wireless Laboratory Environment

Previous results:

- Complexity measurements taken offline
- Adapted modulation/forward error-correction to Rayleigh channel
- Significant computational savings (5 fold)
- Marginal throughput degradation

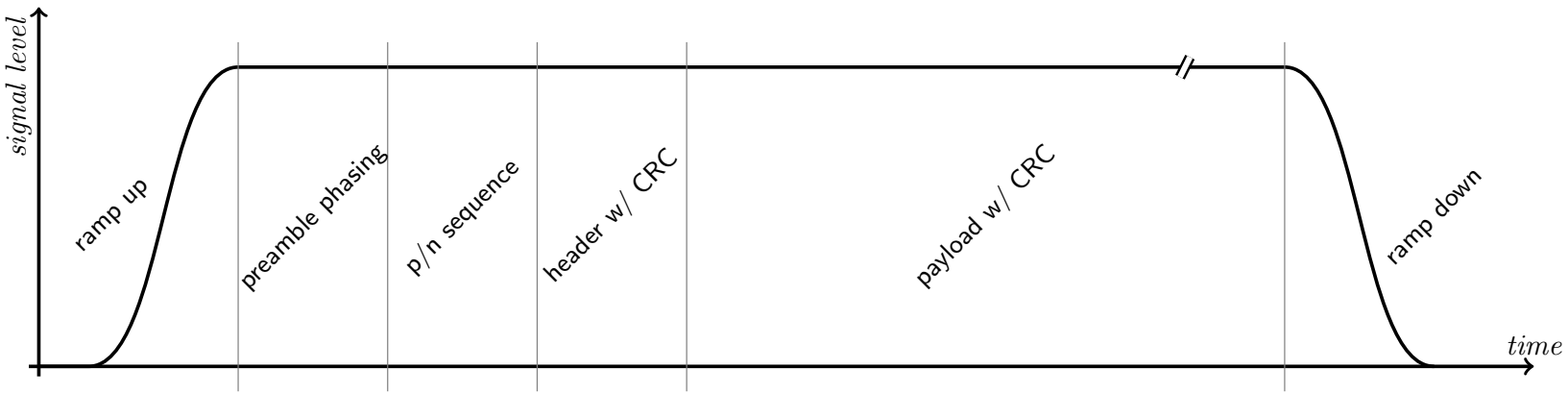
Build on previous simulation results

- Need more convincing results, rather than just simulations
- Need to observe entire receiver chain

Framing Structure

Developed unique framing structure

- Narrowband linear modulation
- Capable of fast synchronization
- Completely adjustable payload
- Embedded framing descriptor (receiver auto-configures)

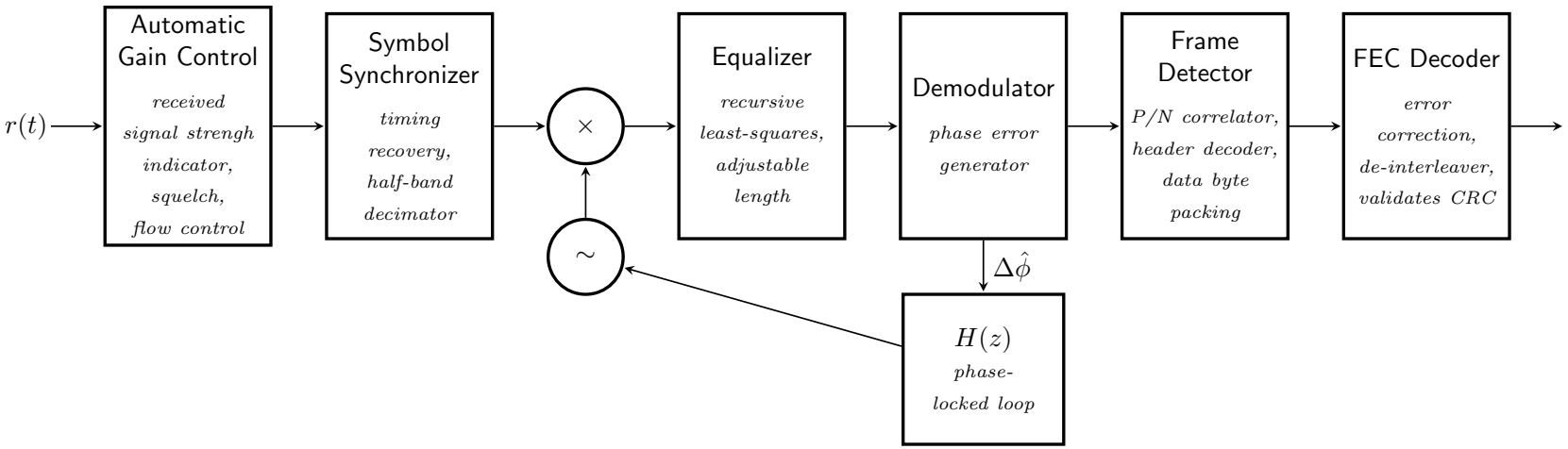


Transceiver description

Receiver consists of

- automatic gain control with squelch
- RSSI and SNR estimations
- NCO with 2^{nd}-order PLL for carrier recovery
- multirate decimator for symbol timing recovery
- frame lock detection
- variable-level demodulator (PSK, QAM, APSK, etc. with 1–8 bits/symbol)
- FEC decoder
- cyclic redundancy check (data integrity validation)

Receiver System Diagram



Experimentation

mod/fec pairs

Modulation schemes

PSK : 2,4,8,16,32

QAM : 4,8,16,32,64,128,256

Experimentation

mod/fec pairs

Modulation schemes

PSK : 2,4,8,16,32

QAM : 4,8,16,32,64,128,256

Forward error-correction schemes:

$\mathcal{K}^{(-)}$: none, Hamming (12,8)

$\eta^{(+)}$: convolutional (Viterbi), $K = 9$, rate $(\frac{1}{2}, \ldots, \frac{7}{8})$

Experimentation

mod/fec pairs

Modulation schemes

PSK : 2,4,8,16,32

QAM : 4,8,16,32,64,128,256

Forward error-correction schemes:

$\mathcal{K}^{(-)}$: none, Hamming (12,8)

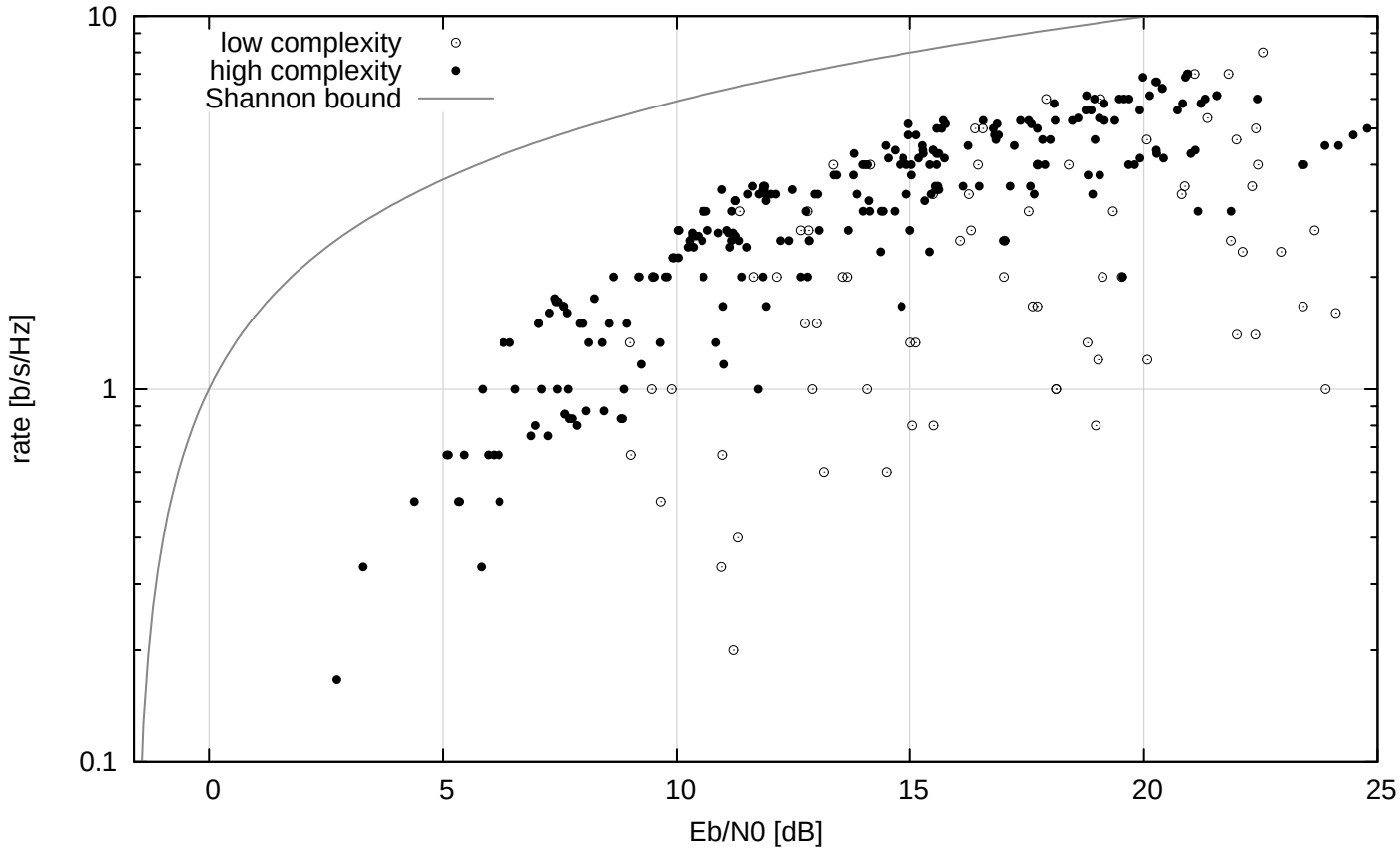
$\eta^{(+)}$: convolutional (Viterbi), $K = 9$, rate $(\frac{1}{2}, \ldots, \frac{7}{8})$

Two sets of mod/fec pairs were chosen

- Spectral efficiency-driven: $\mathcal{S}_\eta$
- Computational complexity-driven: $\mathcal{S}_\mathcal{K}$

Experimentation

mod/fec pairs



mod/fec pairs

spectral efficiency-driven ($\mathcal{S}_\eta$)

Mod. scheme	FEC scheme	$\eta, [b/s/Hz]$	$\mathcal{K}_i, [kcycles/bit]$
QPSK	conv. $r = 1/3$, $K = 9$	0.667	12.512
QPSK	conv. $r = 2/3$, $K = 9$	1.333	8.135
QPSK	conv. $r = 3/4$, $K = 9$	1.500	7.568
QPSK	conv. $r = 7/8$, $K = 9$	1.750	7.064
16-QAM	conv. $r = 2/3$, $K = 9$	2.667	6.138
16-QAM	conv. $r = 3/4$, $K = 9$	3.000	5.733
16-QAM	conv. $r = 4/5$, $K = 9$	3.200	5.753
16-QAM	conv. $r = 7/8$, $K = 9$	3.500	5.281
64-QAM	conv. $r = 2/3$, $K = 9$	4.000	5.254
64-QAM	conv. $r = 3/4$, $K = 9$	4.500	4.975
64-QAM	conv. $r = 4/5$, $K = 9$	4.800	4.840
64-QAM	conv. $r = 7/8$, $K = 9$	5.250	4.831

mod/fec pairs

computational efficiency-driven ($\mathcal{S}_{\mathcal{K}}$)

Mod. scheme	FEC scheme	$\eta, [b/s/Hz]$	$\mathcal{K}_i, [kcycles/bit]$
BPSK	Hamming $(12, 8)$	0.667	7.874
QPSK	Hamming $(12, 8)$	1.333	3.239
QPSK	uncoded	2.000	3.013
8-PSK	uncoded	3.000	2.264
16-QAM	uncoded	4.000	1.938
32-QAM	uncoded	5.000	1.474

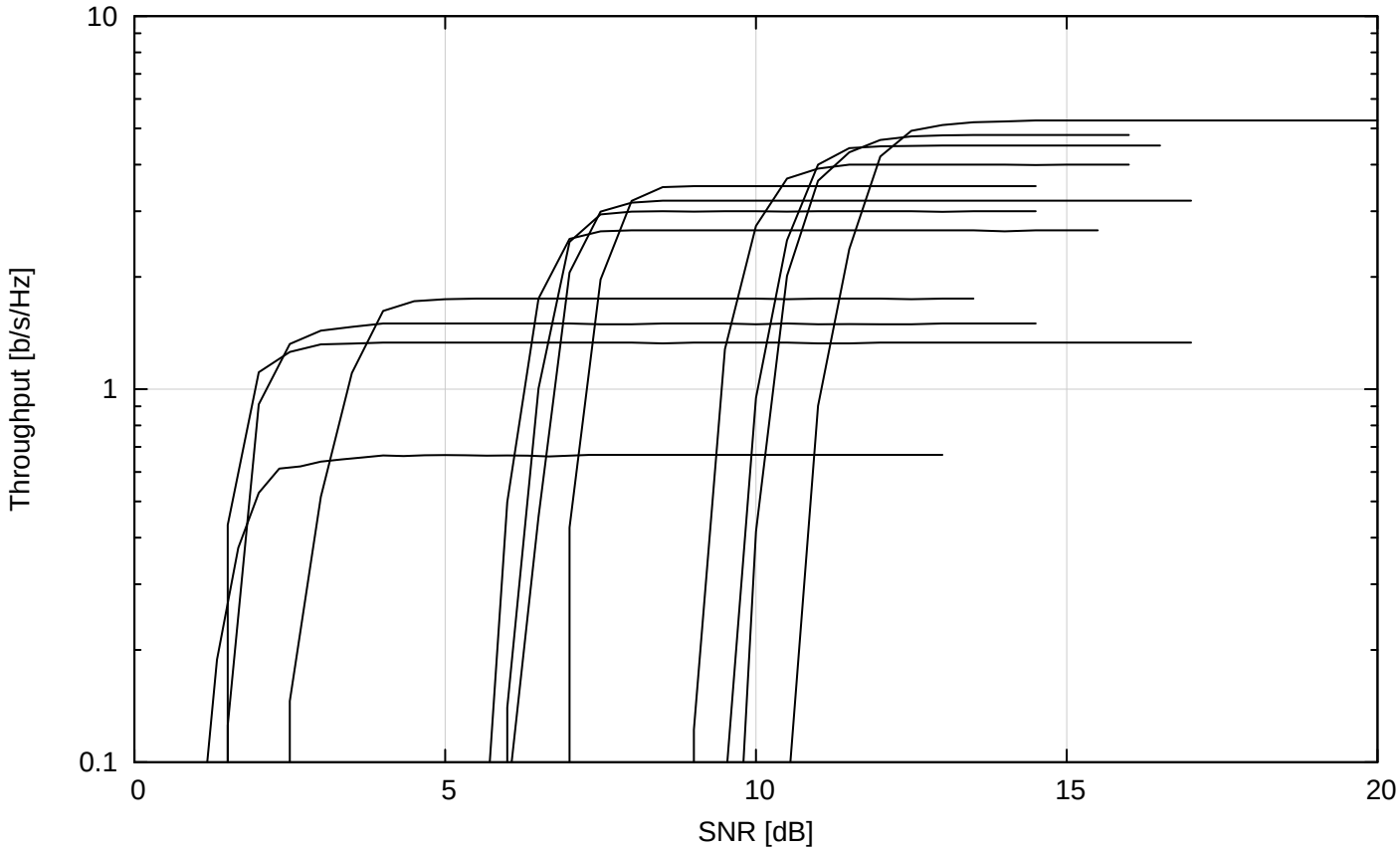
Experimentation

Wireless link established between two computers

- RF hardware: two USRPs
- 462MHz carrier
- 225kHz symbol rate
- receiver: Intel Pentium processor
- randomly-varying transmit power
- received SNR ranges from ∼0dB through ∼20dB

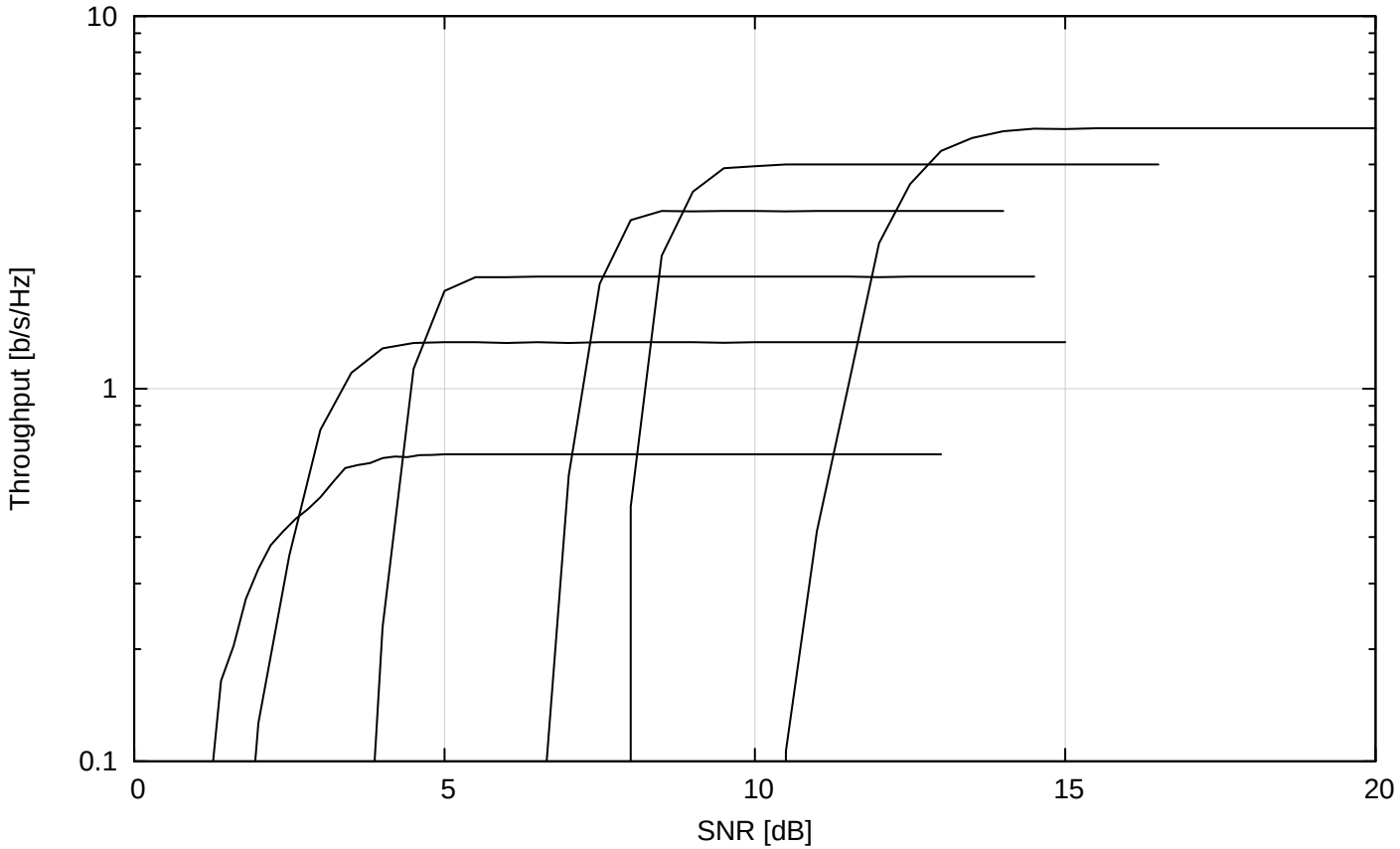
Results (over the air)

average spectral efficiency, $\bar{\eta}$ [b/s/Hz], $\mathcal{S}_\eta$



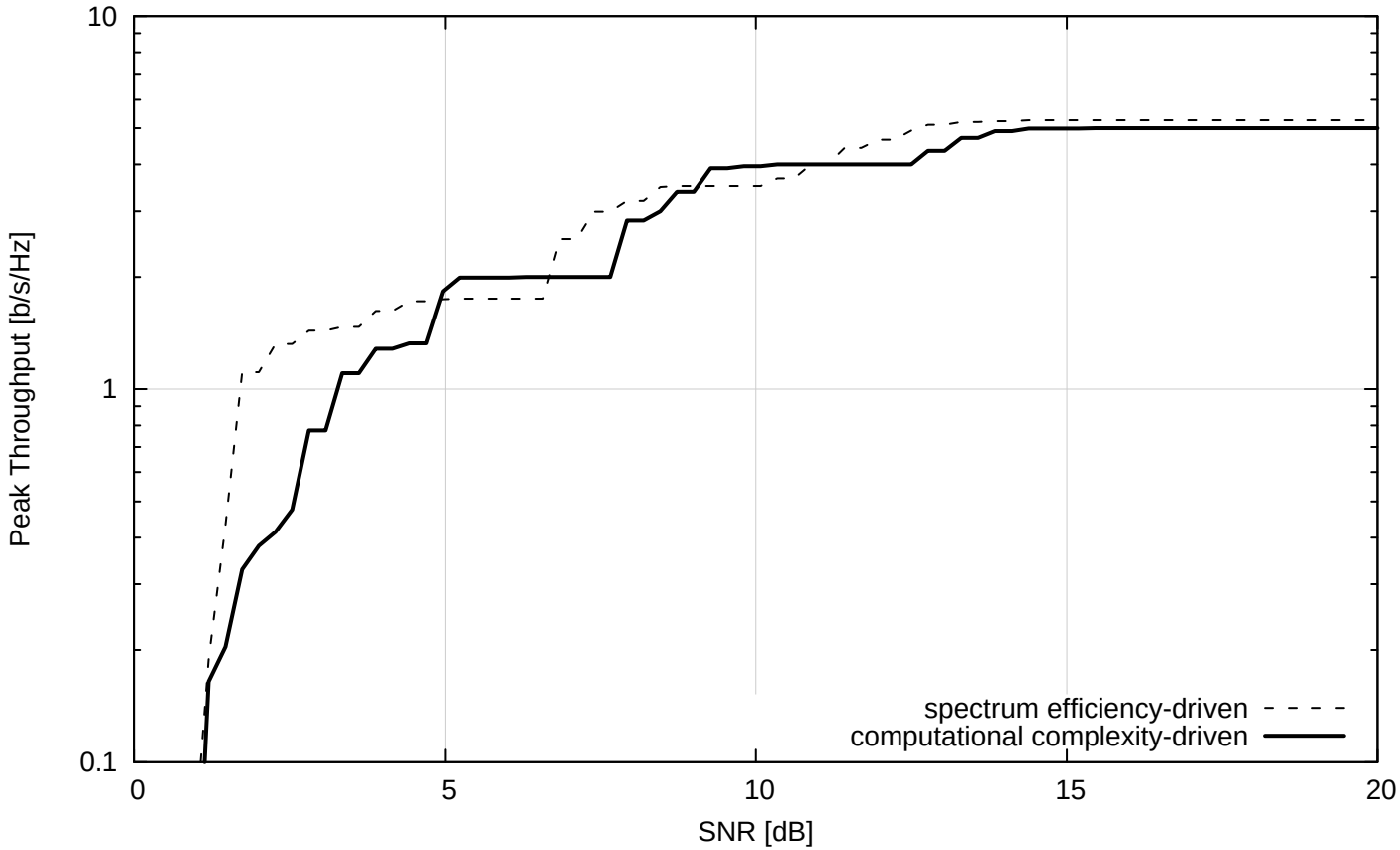
Results (over the air)

average spectral efficiency, $\bar{\eta}$ [b/s/Hz], $\mathcal{S}_{\mathcal{K}}$



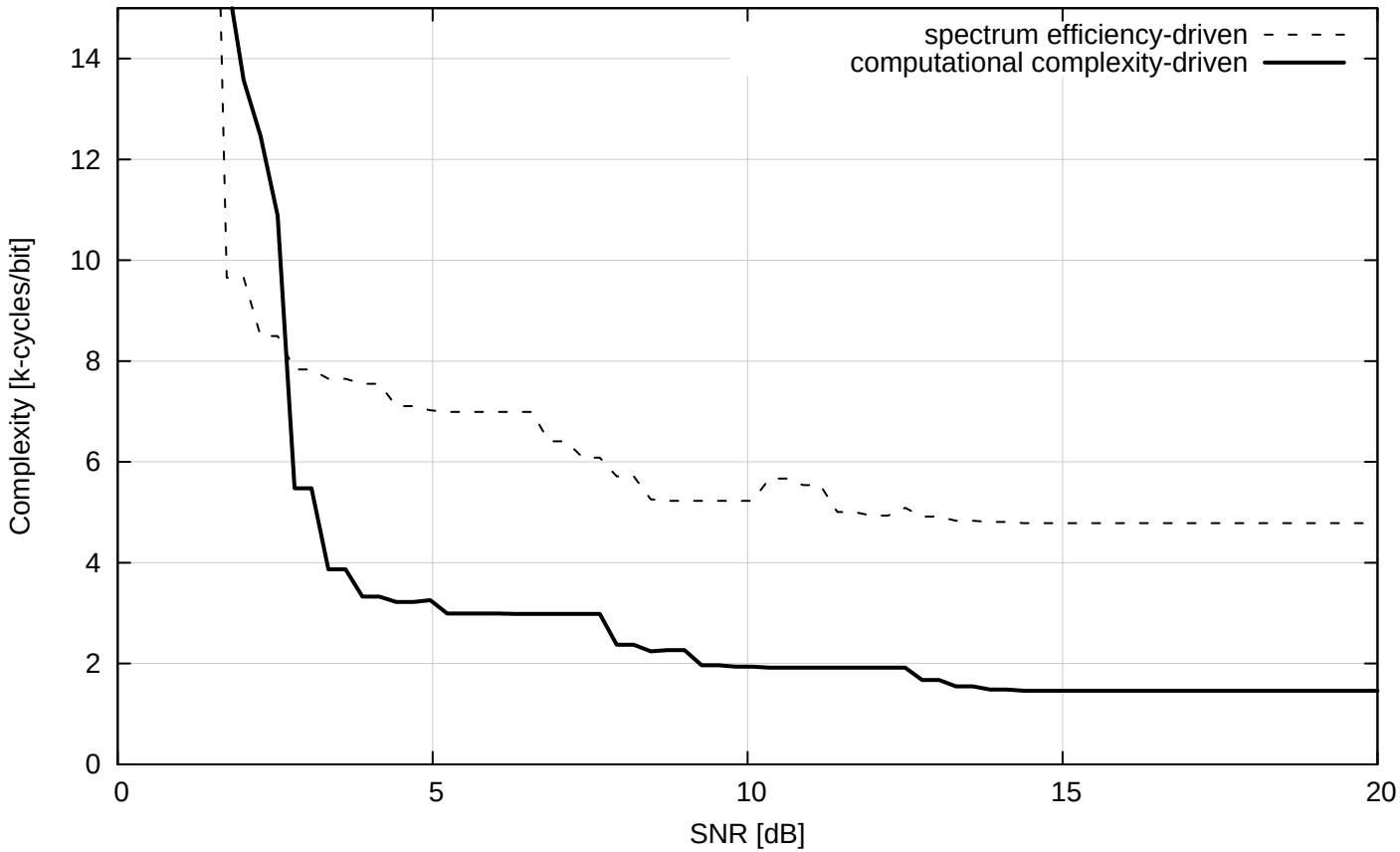
Results (over the air)

peak spectral efficiency, $\bar{\eta}$ [b/s/Hz] (comparison)



Results (over the air)

average computational complexity, $\bar{\mathcal{K}}$ [kcycles/bit]



Observations

The following observations may be made from the figures:

- Spectral efficiency, $\bar{\eta}$, of set 1 is *nearly* always better than set 2 (expected)

Observations

The following observations may be made from the figures:

- Spectral efficiency, $\bar{\eta}$, of set 1 is *nearly* always better than set 2 (expected)
- Largest discrepancy, however, is less than $\frac{1}{2}$ b/s/Hz

Observations

The following observations may be made from the figures:

- Spectral efficiency, $\bar{\eta}$, of set 1 is *nearly* always better than set 2 (expected)
- Largest discrepancy, however, is less than $\frac{1}{2}$ b/s/Hz
- Computational complexity, $\bar{\mathcal{K}}$, of set 2 is *nearly* always better than set 1 (expected)

Observations

The following observations may be made from the figures:

- Spectral efficiency, $\bar{\eta}$, of set 1 is *nearly* always better than set 2 (expected)
- Largest discrepancy, however, is less than $\frac{1}{2}$ b/s/Hz
- Computational complexity, $\bar{\mathcal{K}}$, of set 2 is *nearly* always better than set 1 (expected)
- For high average SNR situations, difference is 4 fold

Observations

The following observations may be made from the figures:

- Spectral efficiency, $\bar{\eta}$, of set 1 is *nearly* always better than set 2 (expected)
- Largest discrepancy, however, is less than $\frac{1}{2}$ b/s/Hz
- Computational complexity, $\bar{\mathcal{K}}$, of set 2 is *nearly* always better than set 1 (expected)
- For high average SNR situations, difference is 4 fold
- For low average SNR situations throughput suffers (so does efficiency)

Observations

The following observations may be made from the figures:

- Spectral efficiency, $\bar{\eta}$, of set 1 is *nearly* always better than set 2 (expected)
- Largest discrepancy, however, is less than $\frac{1}{2}$ b/s/Hz
- Computational complexity, $\bar{\mathcal{K}}$, of set 2 is *nearly* always better than set 1 (expected)
- For high average SNR situations, difference is 4 fold
- For low average SNR situations throughput suffers (so does efficiency)
- Using a larger/optimized subset of all pairs could produce even better results

Observations

The following observations may be made from the figures:

- Spectral efficiency, $\bar{\eta}$, of set 1 is *nearly* always better than set 2 (expected)
- Largest discrepancy, however, is less than $\frac{1}{2}$ b/s/Hz
- Computational complexity, $\bar{\mathcal{K}}$, of set 2 is *nearly* always better than set 1 (expected)
- For high average SNR situations, difference is 4 fold
- For low average SNR situations throughput suffers (so does efficiency)
- Using a larger/optimized subset of all pairs could produce even better results
- Simpler receiver could choose to run faster or save power

Summary

In this presentation:

- Provided a simple metric for resource efficiency in SDR
- Provided a simple model for monitoring resource consumption
- Demonstrated energy-quality tradeoff through experimentation
  - adaptive modulation/coding schemes
  - slowly-varying fading channels
  - over-the-air validation of simulated results

Conclusions

- There exists a fundamental tradeoff between algorithm complexity and performance

Conclusions

- There exists a fundamental tradeoff between algorithm complexity and performance
- The provided example, while trivial, demonstrates this fundamental tradeoff

Conclusions

- There exists a fundamental tradeoff between algorithm complexity and performance
- The provided example, while trivial, demonstrates this fundamental tradeoff
- Code optimization only gives you so much

Conclusions

- There exists a fundamental tradeoff between algorithm complexity and performance
- The provided example, while trivial, demonstrates this fundamental tradeoff
- Code optimization only gives you so much
- Complexity performance sensitive to channel conditions

Conclusions

- There exists a fundamental tradeoff between algorithm complexity and performance
- The provided example, while trivial, demonstrates this fundamental tradeoff
- Code optimization only gives you so much
- Complexity performance sensitive to channel conditions
- Significant computational savings can be gained at a reasonable expense

Conclusions

- There exists a fundamental tradeoff between algorithm complexity and performance
- The provided example, while trivial, demonstrates this fundamental tradeoff
- Code optimization only gives you so much
- Complexity performance sensitive to channel conditions
- Significant computational savings can be gained at a reasonable expense
- Tip of the iceberg: consider also filtering, decimation, mixing, conversion...