ECE 598NSG/498NSU
Deep Learning in Hardware
Fall 2020

Finite-precision Dot Products

Naresh Shanbhag

Department of Electrical and Computer Engineering

University of Illinois at Urbana-Champaign

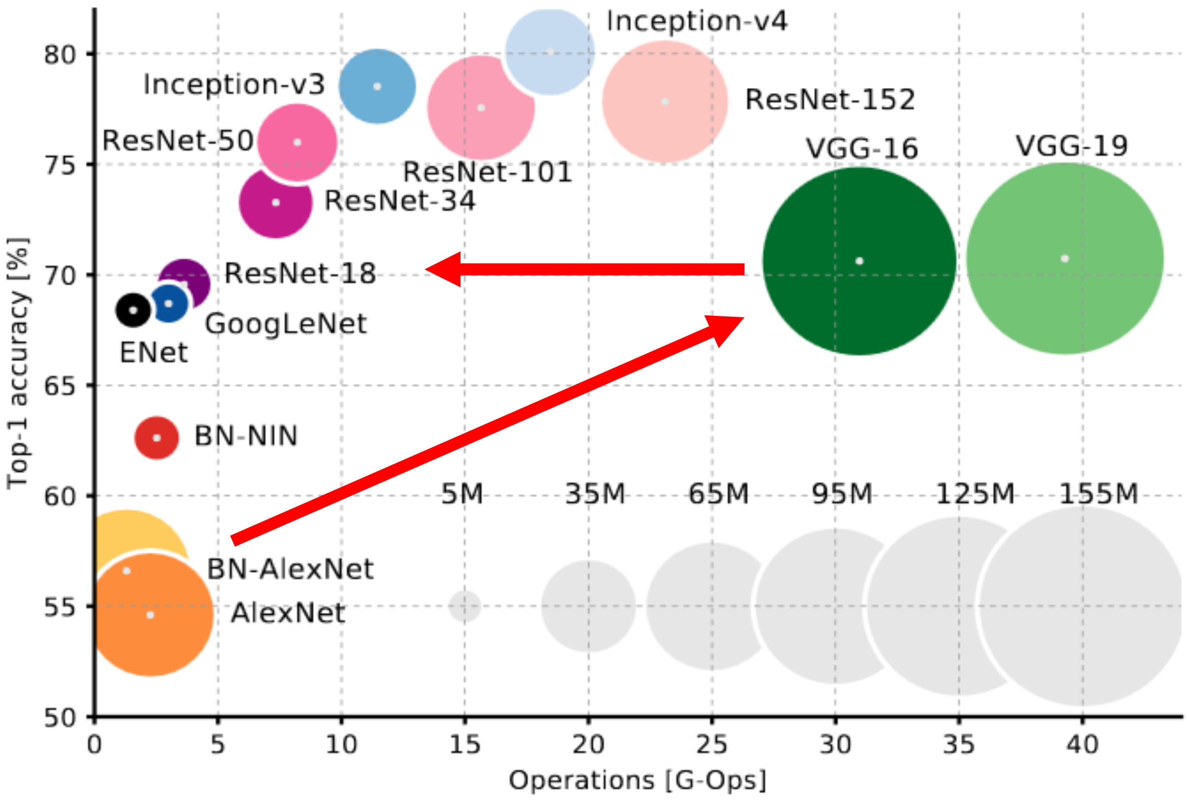
http://shanbhag.ece.uiuc.edu

Today

- quantization theory
- number representations & 2's complement arithmetic
- fixed-point dot products

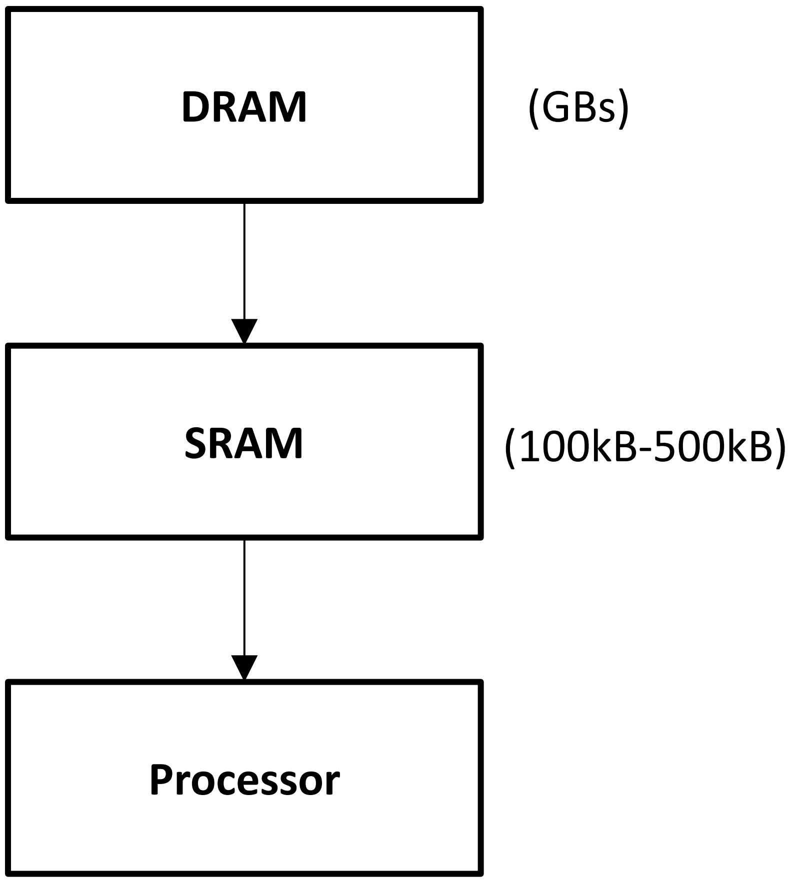
Accuracy vs. Complexity Trade-off in DNNs

[Canziani et al., arXiv2016]



- AlexNet came first – can be thought of as baseline
- VGG-Net achieved high accuracy *at the cost of complexity (storage & compute)*
- GoogleNet and ResNet achieved *high accuracy* while maintaining *moderate complexity*

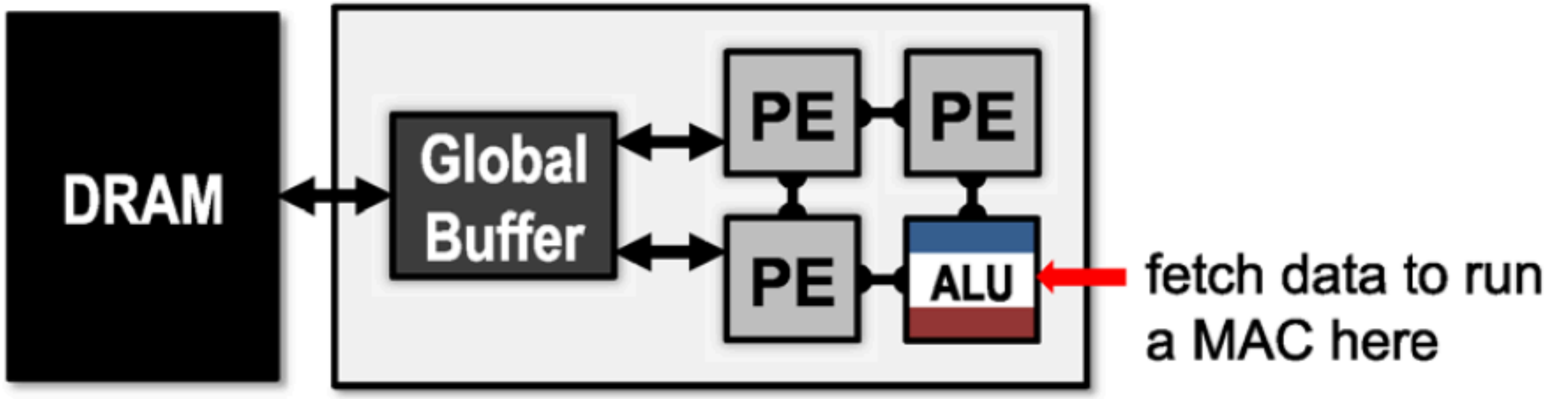
Storage Challenge

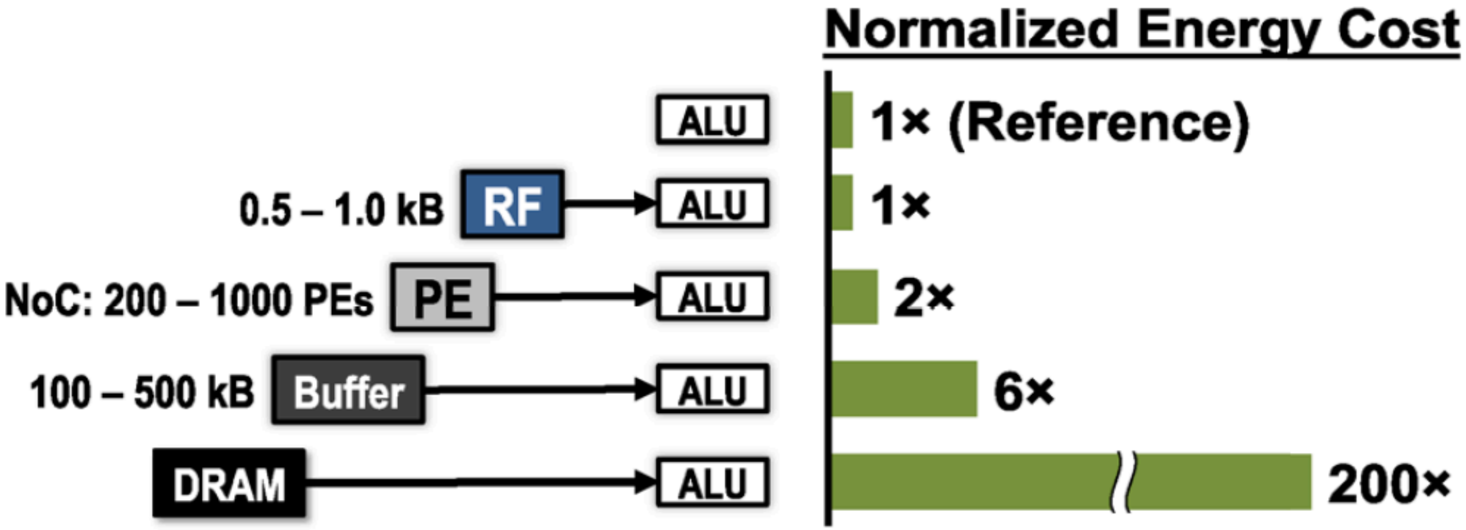


- AlexNet – 63M weights
- 8b/weight → 63MB storage requirements
- overwhelms current on-chip SRAM capacity
- this is inference only – not training

Energy Challenge

[Sze, IEEE Proceedings, December 2017]





- Large model sizes imply a data movement problem:

  DRAM→ SRAM→ PE

- energy and latency costs amplified when data resides far from compute

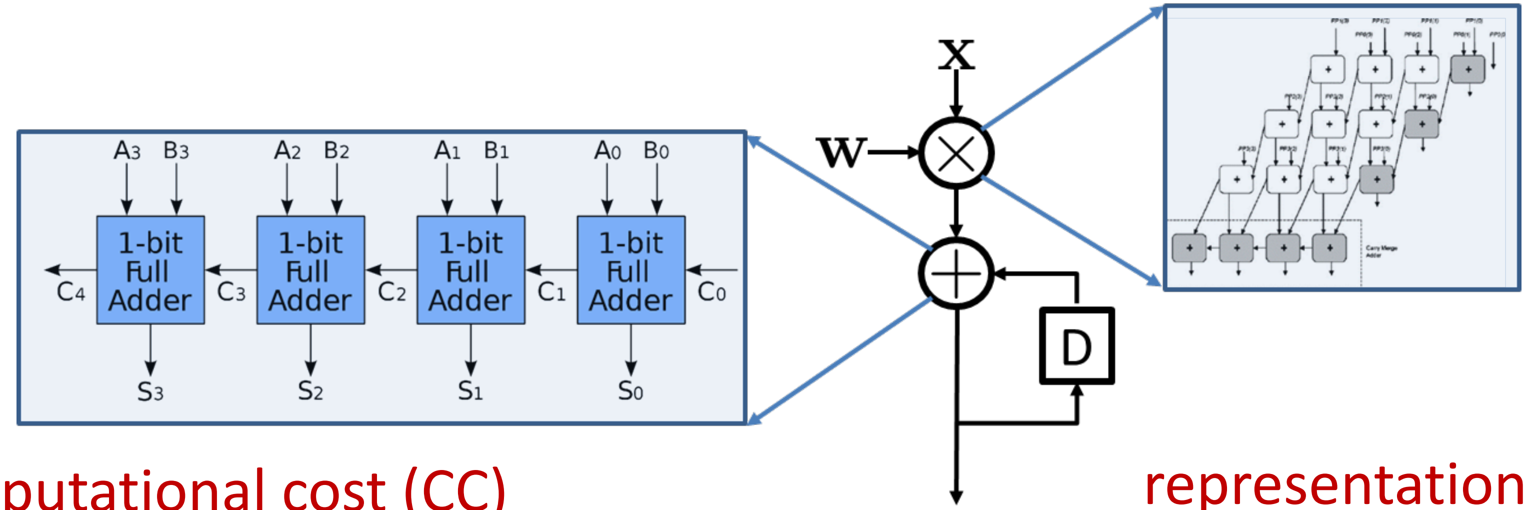
Computing DNNs in Finite-Precision

- precision reduction is a powerful knob for reducing storage and computational requirements

- Cannot reduce precision arbitrarily:
  - reducing precision impacts inference accuracy
  - some variables are more important than others
  - impact of quantization depends on signal distribution
- Next:
  - quantization of variables
  - number representations
  - fixed-point dot-product

Hardware Complexity Metrics

easy to compute from an algorithmic description



computational cost (CC)

(# of 1-b FAs)

representational cost (RC)

(# of bits of storage)

# of dot products

dot product dimension

$$\sum_{l=1}^{L} N_l \left(D_l B_l^{(a)} B_l^{(w)} + (D_l - 1)\left(B_l^{(a)} + B_l^{(w)} + \log_2 D_l - 1\right)\right)$$

activation precision

weight precision

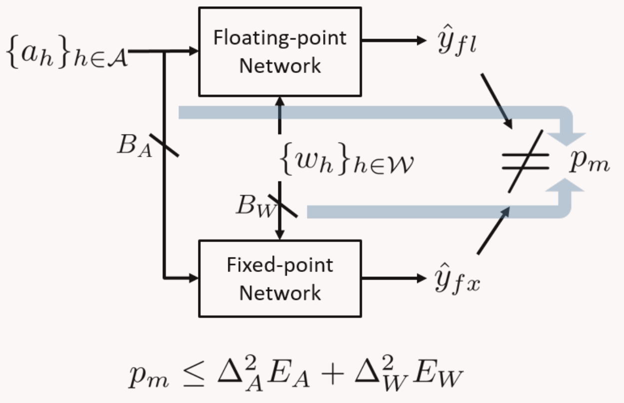
# of weights

$$\sum_{l=1}^{L} \left(R_l^{(a)} B_l^{(a)} + R_l^{(w)} B_l^{(w)}\right)$$

# of activations

Recent UIUC Work – Finite Precision Analysis of DNNs

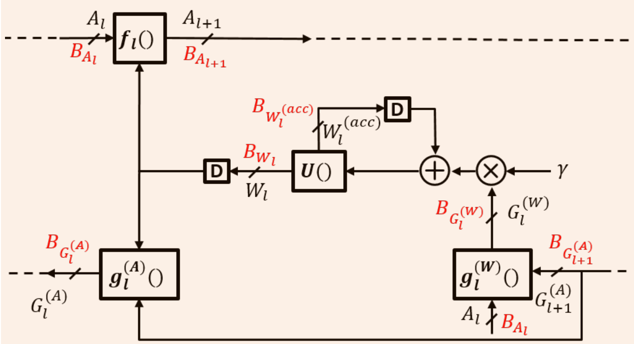
fixed-point inference with theoretical guarantees



[Sakr, Kim, Shanbhag, **ICML 2017**]

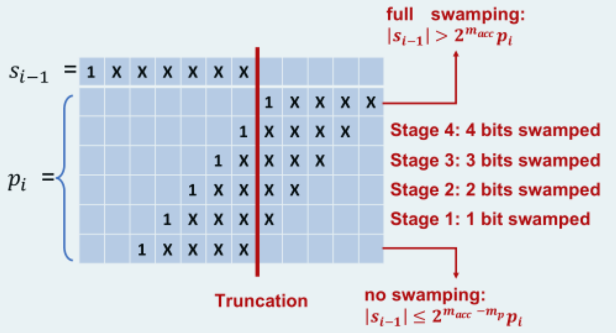
[Sakr & Shanbhag, **ICASSP 2018**]

true fixed-point training with close-to-minimal precision



[Sakr & Shanbhag, **ICLR 2019**]

floating-point training with accumulation bit-width scaling



[Sakr, Shanbhag, **ICLR 2019**]

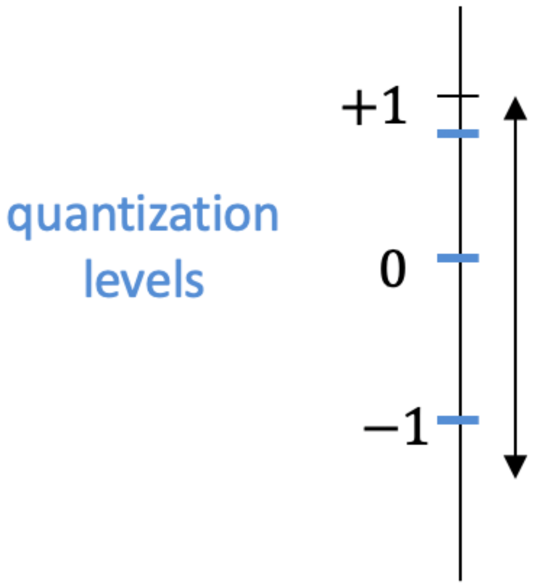
(with IBM: K. Gopalakrishnan,

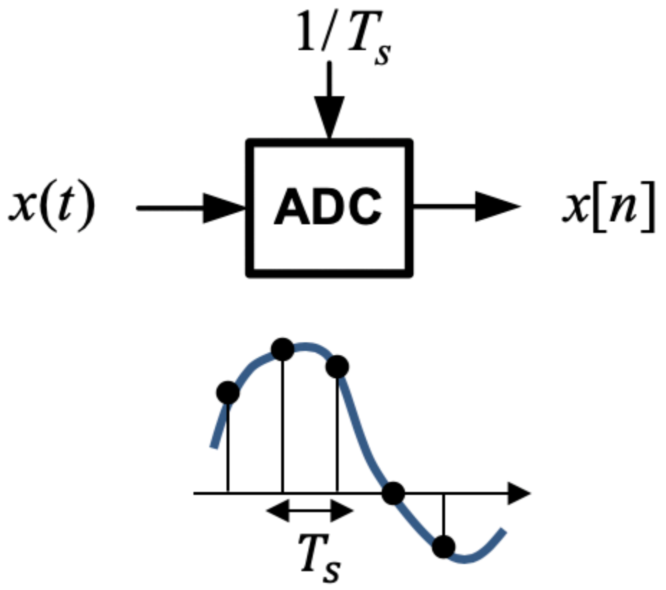
N. Wang, C.-Y. Chen, A. Agrawal,J. Choi,)

Quantization

Quantization

- The process of obtaining using a finite number of discrete levels to represent a continuous-valued variable is called quantization
- Example: an analog-to-digital converter (ADC) (ignore time index n)

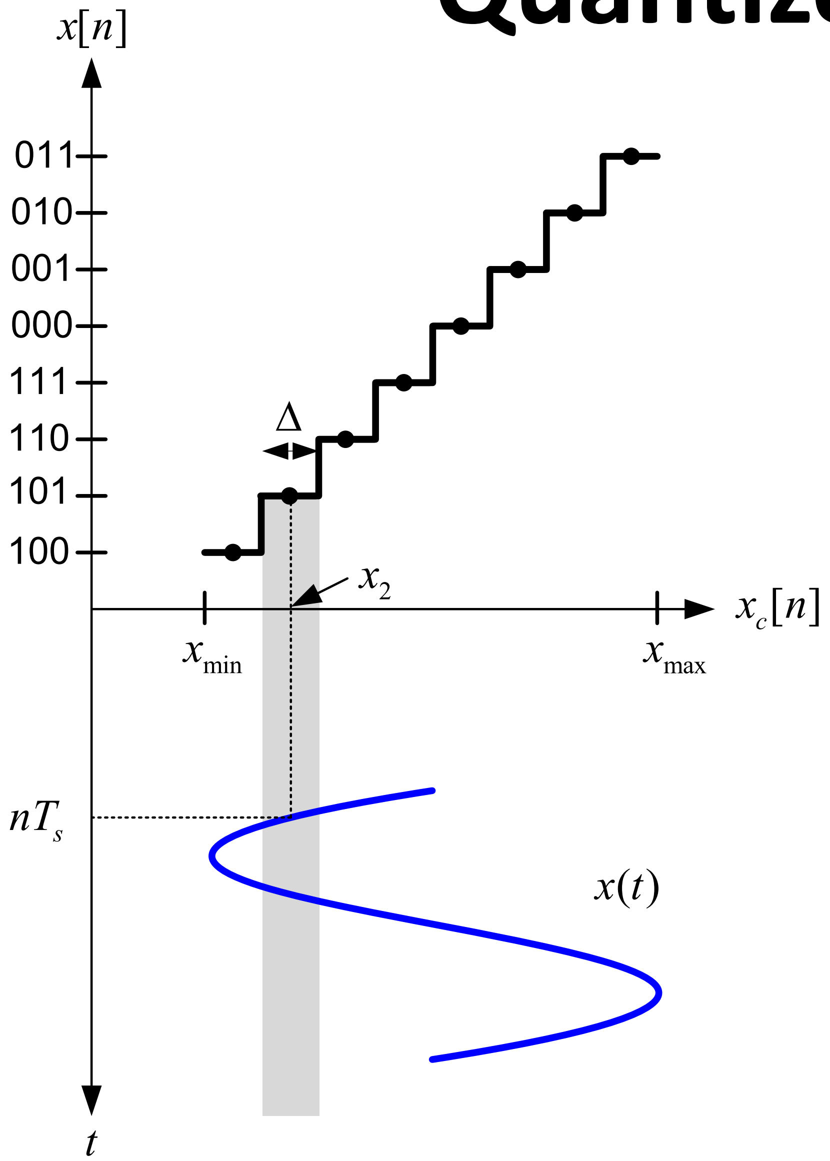




code

+0.75→0.11
0.00→ 0.00
-1.00→ 1.00

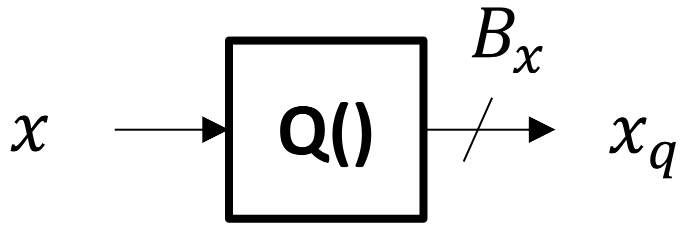
Quantizer Staircase Model



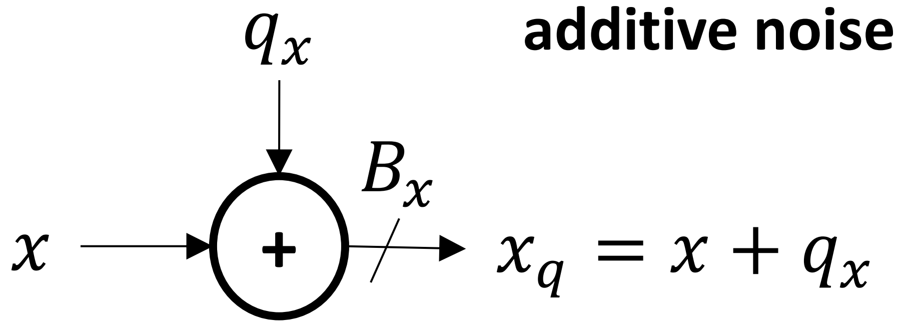
- quantizer can also be described by its input-to-output mapping
- useful for simulating quantizers
- this mapping is parameterized in terms of the *step-sizes* Δ_i and the *quantization levels* r_i $(i = 1, \ldots, 2^{B_x})$
- r_i's have a digital code associated with it

Additive Quantization Noise Model

quantizer symbol



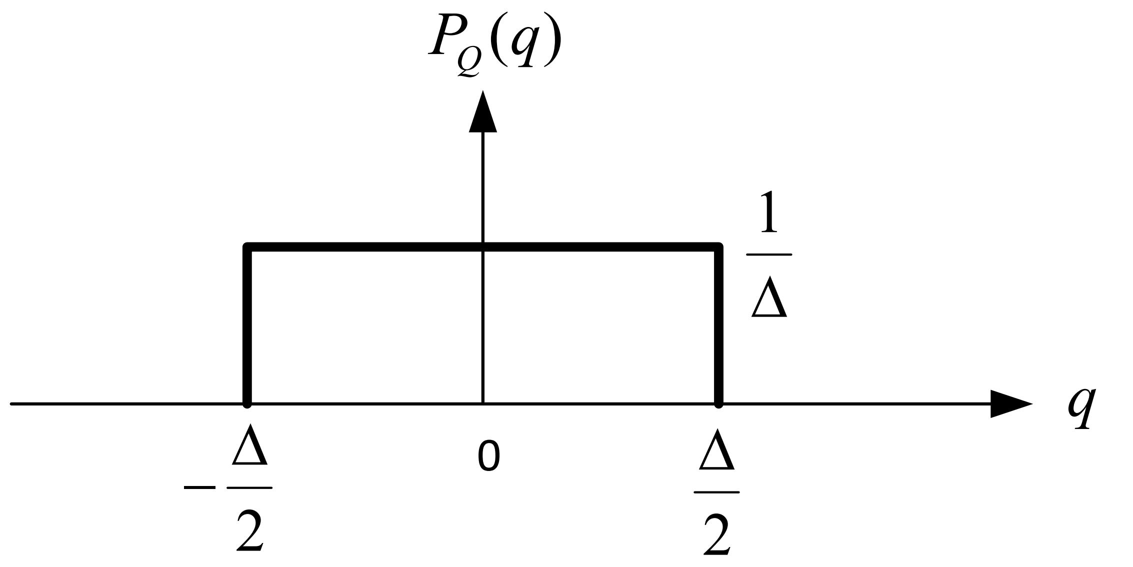
additive noise model



- x : floating-point or analog valued or infinite-precision scalar data
- q_x: quantization noise in x
- $x_q = Q[x]$: quantized value of x
- additive model: q_x is assumed to be independent of x

A Useful Result

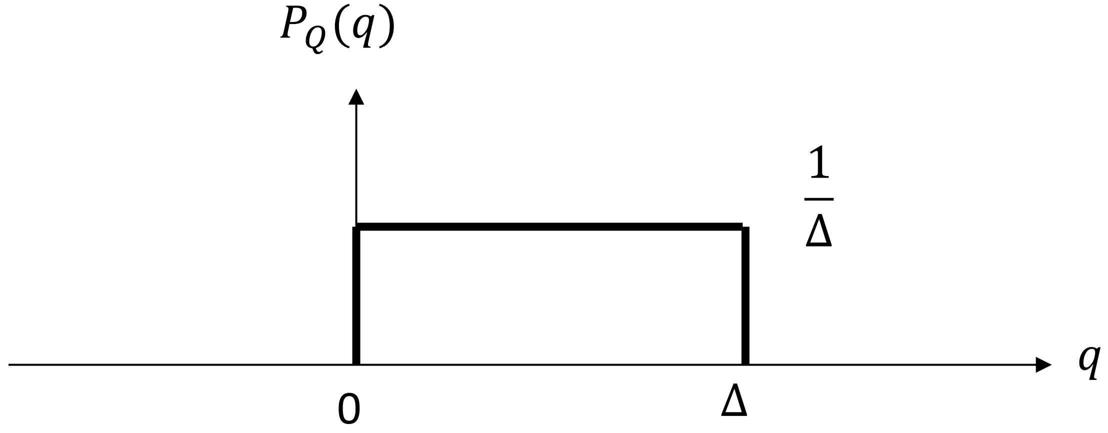
probability density function of q



- If Q is a uniformly distributed in $[-\frac{\Delta}{2}, +\frac{\Delta}{2}]$ then $\mu_Q = 0; \sigma_Q^2 = \frac{\Delta^2}{12}$
- quantization noise is often well modeled as a uniformly distributed RV

Quantization Noise - Truncation

- assuming quantization error is due to truncation:

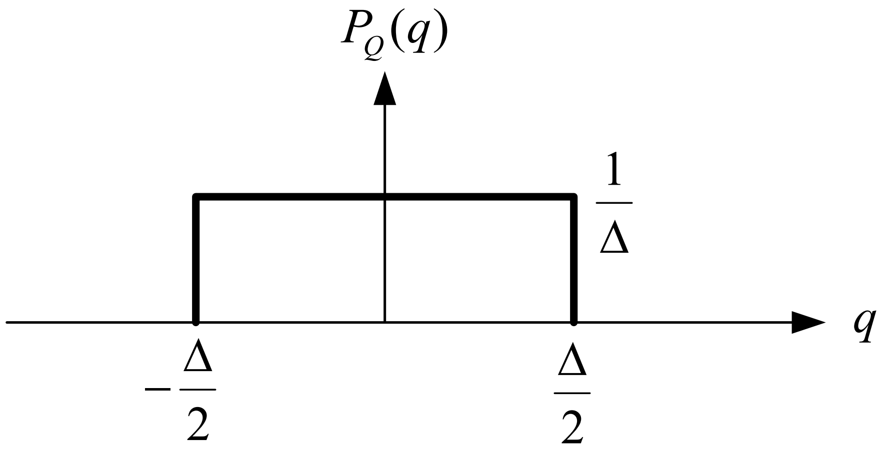


where $\Delta = \frac{2}{2^{B_x}} = 2^{-(B_x-1)}$ and hence (assumes $|x| \leq 1$)

$$\sigma_{q_x}^2 = \frac{2^{-2B_x}}{3} = \frac{\Delta^2}{12}$$

Quantization Noise – Round-off

- assuming quantization error is due to round-off:



where $\Delta = \frac{2}{2^{B_x}} = 2^{-(B_x-1)}$ and hence (assumes $|x| \le 1$)

$$\sigma_{q_x}^2 = \frac{2^{-2B_x}}{3} = \frac{\Delta^2}{12}$$

- same noise variance as truncation but with zero mean but needs more computation

Measuring Quantizer Accuracy

- Signal-to-quantization noise ratio (SQNR) measures the accuracy of the quantizer

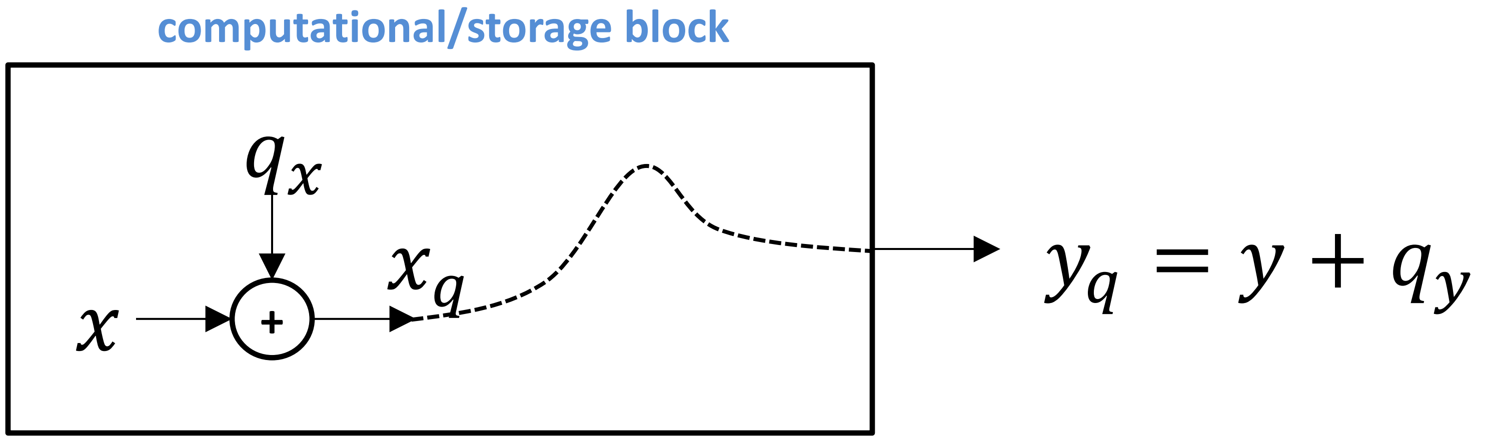
$$SQNR = 10\log_{10}\left(\frac{\sigma_x^2}{\sigma_q^2}\right)$$

$$q = x - Q[x]$$

- need to treat x as a random variable (X) with a density function $f_X(x)$ - x is a sample of X

- SQNR improves as more bits are assigned to represent X

Quantization Noise Analysis



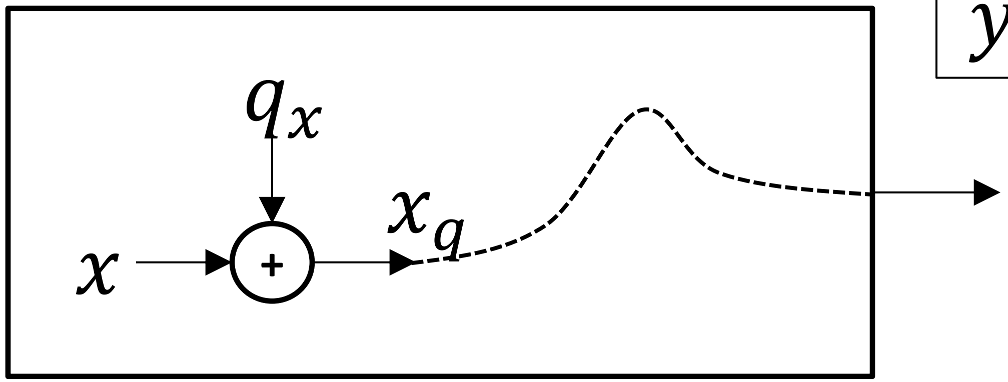
$$SQNR_y = 10\log_{10}\left[\frac{\sigma_y^2}{\sigma_{q_y}^2}\right]$$

$q_y = f(q_x) = \frac{\partial y}{\partial x} q_x$ (first-order term in Taylor series expansion of $f(x)$)

- determine $\sigma_{q_y}^2$ as a function of $\sigma_{q_x}^2$: need to find $q_y = f(q_x)$
- contribution of q_x to output quantization noise q_y: $\sigma_{q(x\rightarrow y)}^2$
- sum all such contributions → total output quantization noise

True SQNR Requirements – Application Dependent

computational/storage block



$$\boxed{y = s + n = f(x)}$$

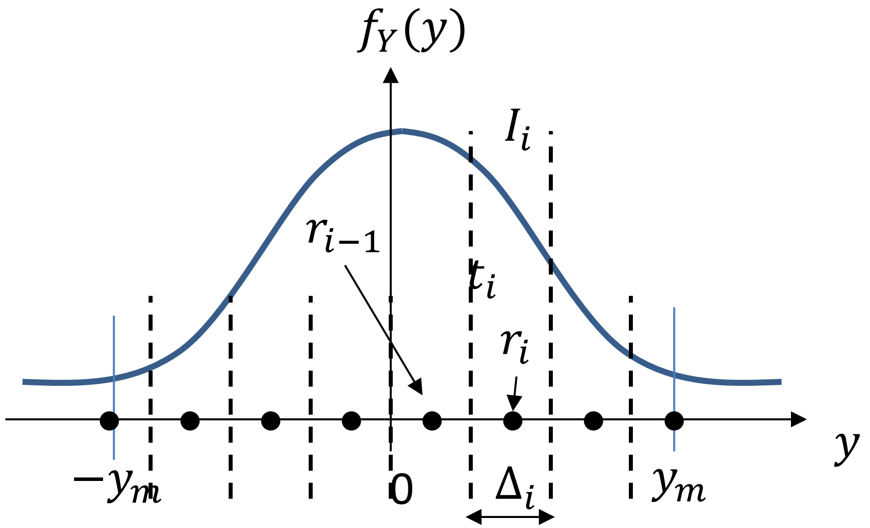
$$y_q = y + q_y$$

- Output SNR (SNR_y) requirements set by application:

$$SNR_y = 10\log_{10}\left[\frac{\sigma_s^2}{\sigma_n^2}\right]$$

- Need to set $SQNR_y \gg SNR_y$, e.g., $SQNR_y = \boldsymbol{SNR_y} + 6dB$
- How to compute SQNR for a dot product?

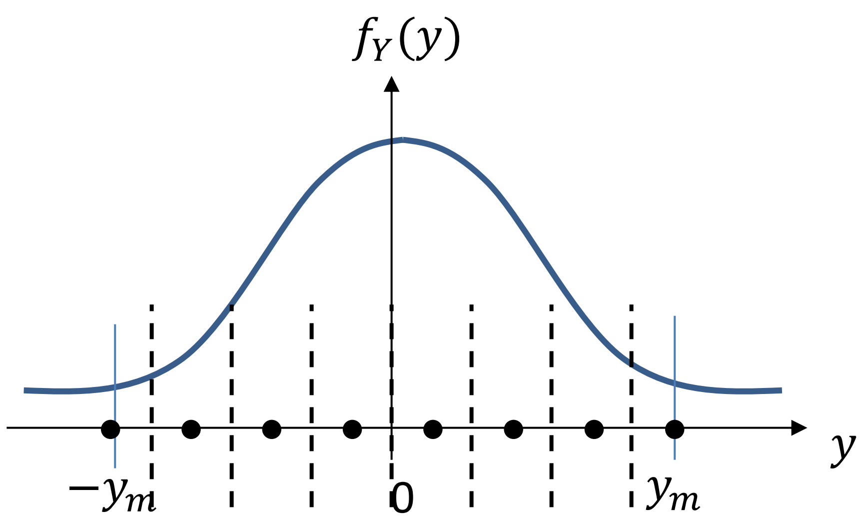
The Quantization Problem



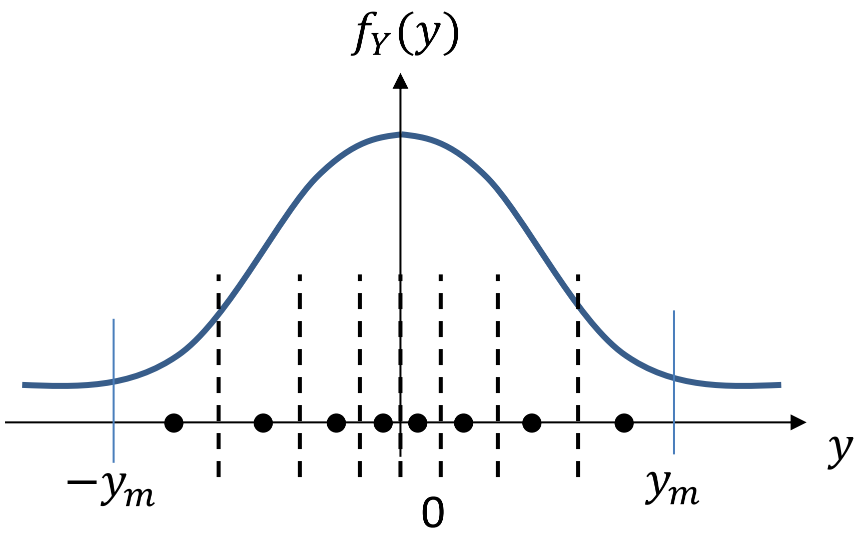
- Δ_i: i^{th} quantizer step-size
- r_i: i^{th} quantizer level
- t_i: i^{th} quantizer threshold
- I_i: i^{th} quantization interval
- B: number of bits
- $M = 2^B$: number of quantization intervals

- SQNR depends on the location and number of reference levels
- General rule – concentrate levels in higher density regions of X
- What are the SQNR-optimal values of Δ_i, r_i, t_i for a given number of bits B?

Uniform vs. Non-uniform Quantization



uniform quantization



non-uniform quantization

- General rule – concentrate levels in higher density regions of y
- Optimum (SQNR sense) quantization → Lloyd-Max Algorithm

Lloyd-Max Quantizer

- algorithm to determine $M = 2^B$ quantization levels $\{r_q\}_{q=0}^{M-1}$ as well as $M - 1$ quantization thresholds $\{t_q\}_{q=1}^{M-1}$
- Step 1: guess initial quantization levels $\{r_q\}_{q=0}^{M-1}$ (assume uniform)
- Step 2: calculate quantization thresholds:

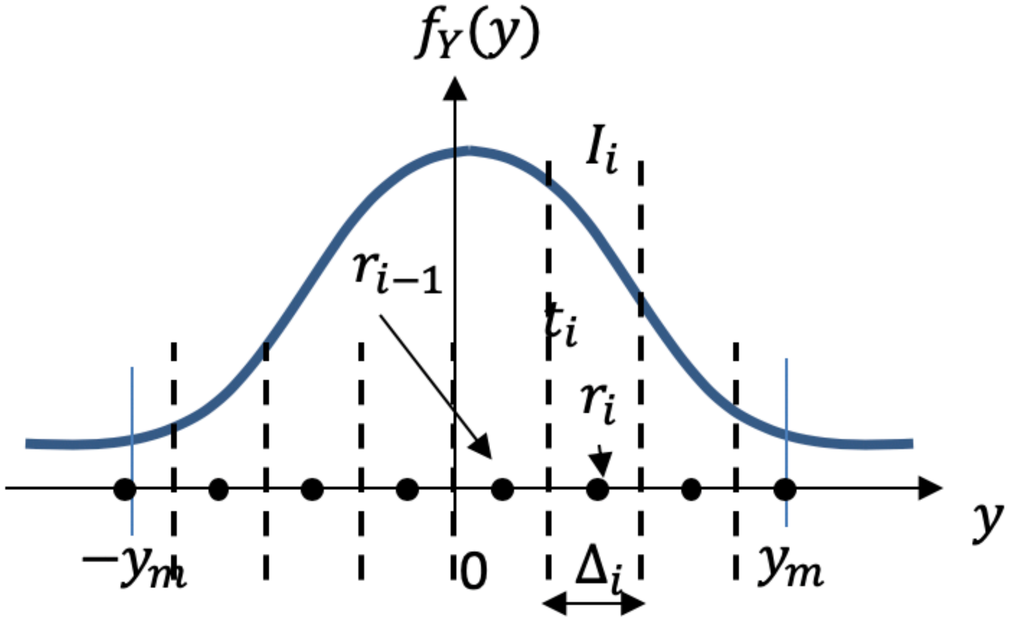
$$t_q = \frac{r_q + r_{q-1}}{2} \quad q = 1,2, \dots, M-1$$

- Step 3: calculate new quantization levels as centroids (conditional mean) of new regions:

$$r_q = \frac{\int_{t_q}^{t_{q+1}} y f_{Y(y)}}{\int_{t_q}^{t_{q+1}} f_{Y(y)}} \quad q = 0,1, \dots, M-1$$

- Step 4: repeat Steps 2 & 3 until r_q and t_q values converge

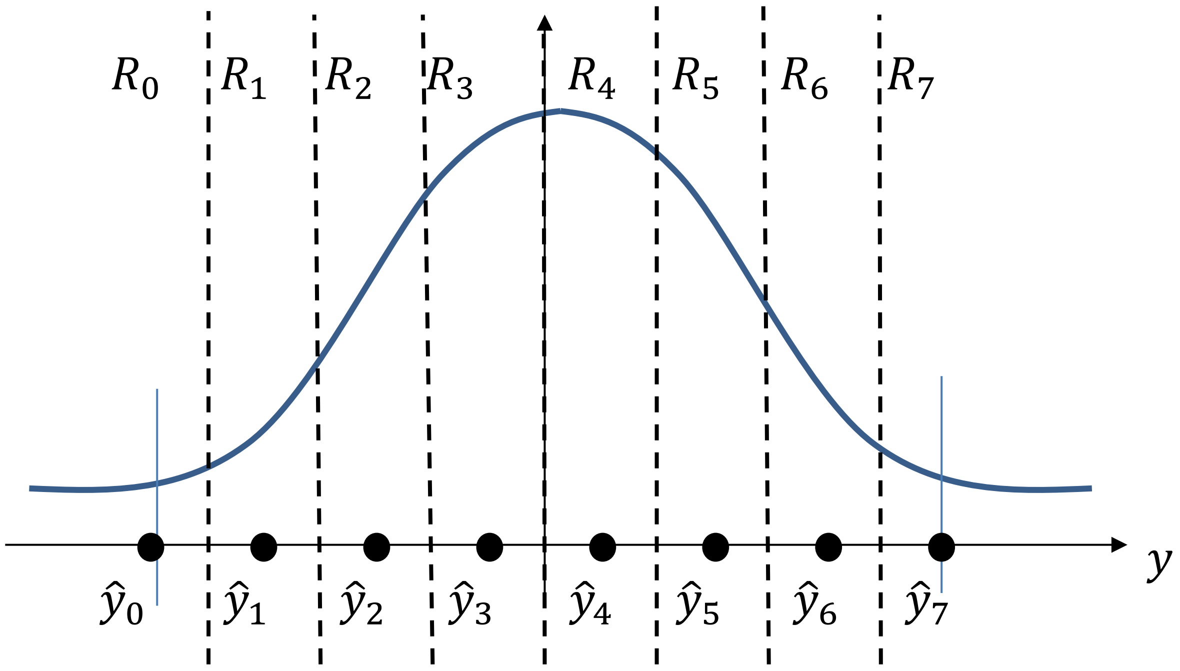
Calculating the SQNR of a Quantizer



$$\Delta = \frac{2y_m}{2^B} = \frac{y_m}{2^{B-1}}$$

- assume quantizer has been designed: r_qs for a given $f_Y(y)$ are known → now calculate the SQNR?

$$\sigma_q^2 = MSE = \boldsymbol{E}[(Y - Q[Y])^2] = \sum_i \int_{I_i} (y - r_i)^2 f_Y(y)dy$$



- once the MSE is computed we can compute the SQNR as

$$SQNR_y = \frac{\sigma_y^2}{\sigma_q^2} = \frac{\boldsymbol{E}[Y^2]}{\boldsymbol{E}[(Y - Q[Y])^2]}$$

SQNR of a Uniform Quantizer

- The SQNR (in dB) of a uniform quantizer is given by

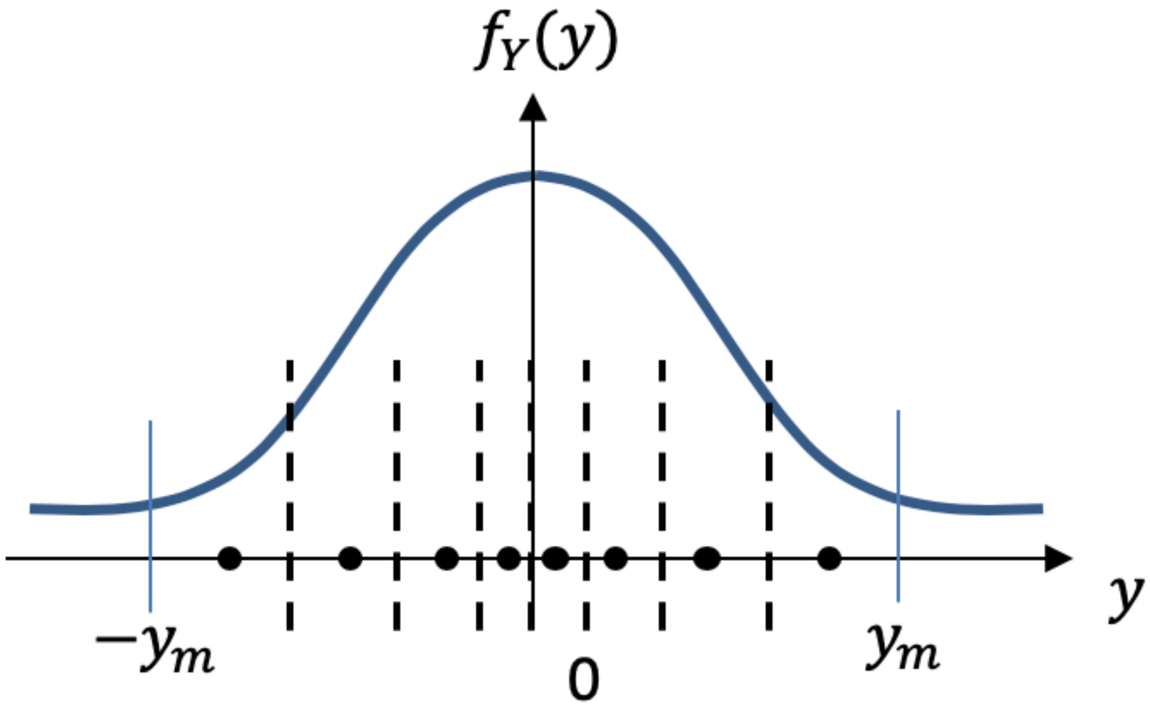
$$SQNR_y(dB) = 6B_y + 4.78 - PAR_y(dB)$$

where $PAR_y = \frac{y_m}{\sigma_y} = \zeta_y$

- Proof: $SQNR_y = \frac{\sigma_y^2}{\sigma_q^2} = \frac{\sigma_y^2}{(\Delta^2/12)}$; where $\Delta = \frac{2y_m}{2^{B_y}}$

- each bit of quantization increases the SQNR by 6dB

Peak to Average (Power) Ratio - PAR

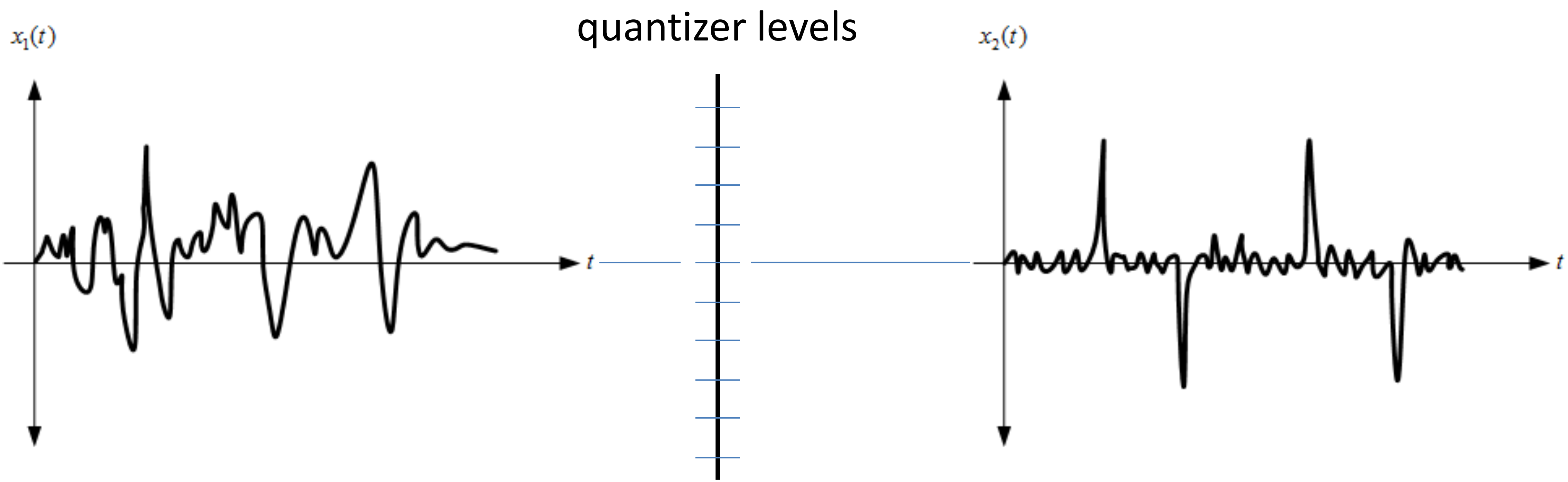


- PAR is the peak-to-average (power) ratio of the signal $y(t)$

$$\zeta_y = \frac{y_m}{\sigma_y}$$

where $-y_m \leq y \leq y_m$ and σ_y^2 is the variance of $y(t)$

$\zeta_{x1} < \zeta_{x2}$



- $\zeta_x = \frac{x_m}{\sigma_x}$
- signal with higher PAR needs higher precision to achieve a target $SQNR$
  - $x_1(t)$ needs fewer bits than $x_2(t)$ to achieve the same $SQNR$

Estimating vs. Evaluating SQNR

- **Evaluating SQNR** → evaluating an expression for SQNR
  - E.g., $SQNR_x(dB) = 6B_x + 4.78 - PAR_x(dB)$
- **Estimating SQNR** → using simulations to empirically calculate SQNR (need sufficiently large number of samples → large N)
  - Step 1: generate samples of X: $x_1, x_2, \ldots\ldots, x_N$
  - Step 2: quantize samples: $Q[x_1], Q[x_2], \ldots. Q[x_N]$ (need quantizer table or staircase mapping)
  - Step 3: calculate quantization noise: $q_1 = x_1 - Q[x_1], q_2 = x_2 - Q[x_2], .., q_N = x_N - Q[x_N]$
  - Step 3: calculate sample variances of X (σ_x^2) and q (σ_q^2) → $SQNR_x = \frac{\sigma_x^2}{\sigma_q^2}$

Example – Evaluating SQNR

- How many bits are needed to obtain an *SQNR* of 43dB for a sinusoidal signal $x[n] = V_m \sin(2\pi f_c t)$?

- Sinusoid peak voltage = V_m

  RMS voltage $= \frac{V_m}{\sqrt{2}}$

  $$PAR_x = 20\log_{10}\left(\frac{V_m}{\frac{V_m}{\sqrt{2}}}\right) = 3dB$$

  $$SQNR = 43 \le 6B_x + 4.8 - 3$$

  $\therefore B_x \ge 6.86 = 7$ bits

- PAR for sinusoidal inputs = 3 dB

Number Representations & 2's Complement Arithmetic

Outline

- floating point representation –
  - FL-$(m+e)$: $x = m \times 2^e$
- fixed-point:
  - FX-m $\equiv$ FL-$(m+0)$: $x = m$
  - 2’s complement, sign-magnitude,…
- logarithmic representation
  - LN-e $\equiv$ FL-$(0+e)$: $x = 2^e$

Fixed-point vs. Floating point

Binarized Neural Networks: Training Neural Networks with Weights and Activations Constrained to +1 or −1

Matthieu Courbariaux*[1] MATTHIEU.COURBARIAUX@GMAIL.COM
Itay Hubara*[2] ITAYHUBARA@GMAIL.COM
Daniel Soudry[3] DANIEL.SOUDRY@GMAIL.COM
Ran El-Yaniv[2] RANI@CS.TECHNION.AC.IL
Yoshua Bengio[1,4] YOSHUA.UMONTREAL@GMAIL.COM

[arxiv, March 2016]

- fixed-point architectures are much less complex (less energy, faster) that floating point ones
- learning algorithms work very well with limited (4b-12b) precision → 1b deep neural networks (BinaryNet) (but why?)
- key questions: how many bits are needed? how to determine it?

Floating-Point Arithmetic

Floating-point number

	sign	exponent			mantissa				
$a =$	0	1	0	0	1	1	0	0	0

$$a = (-1)^{Bs} \times 2^{E} \times (1 + M)$$

Floating-point MAC

$$c \leftarrow c + a \times b$$

a is $(1, e_a, m_a)$ & b is $(1, e_b, m_b)$

- floating-point numbers have three fields: 1 sign bit, e exponent bits, m mantissa bits
- above representation → (1,3,5) – in general $(1, e, m)$

FX and LOG as special cases of FL

Floating-point number

sign exponent mantissa

$a =$

0	1	0	0	1	1	0	0	0

$$a = (-1)^{Bs} \times 2^{E} \times (1 + M)$$

no exponent bits

no mantissa bits

Fixed-point

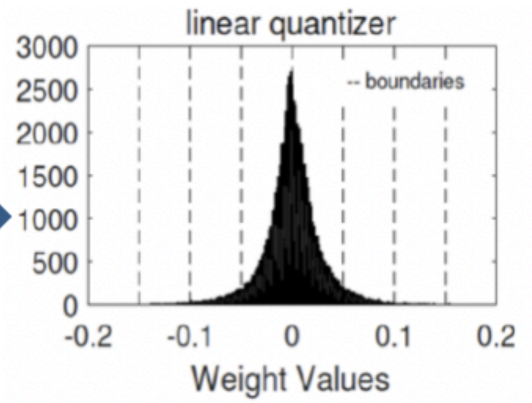
sign mantissa

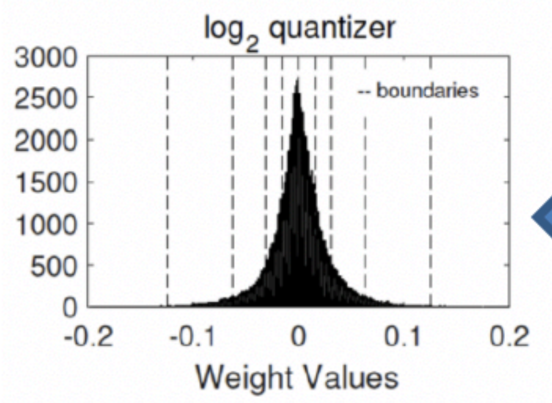
$a =$

0	1	1	0	0	0

$$a = (-1)^{Bs} \times (1 + M)$$

fixed-point number in signed magnitude representation





figures from [Lee et al., ICASSP'17]

log

sign exponent

$a =$

0	1	0	0

$$a = (-1)^{Bs} \times 2^{E}$$

logarithmic quantization

FX Representation: 2's Complement

- B_x bits 2's complement representation of $x[n]$:

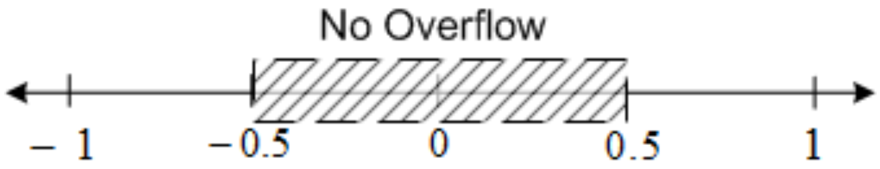
$$x[n] = -b_0 + \sum_{i=1}^{B_x-1} b_i 2^{-i}$$

$b_i \in \{0,1\}$ $\qquad$ b_0 : sign bit

- Assume: $-1 \leq x[n] < 1$ $\rightarrow$ biased towards -ve values;
- no representation for '1' in 2's complement
- compact representation: $[b_0 \cdot b_1 b_2 \ldots b_{B_x-1}]$
  - (.) represents binary point

Addition

- assume $y[n] = x_1[n] + x_2[n]$
- to avoid overflow: $B_y = \max\{B_{x_1}, B_{x_2}\} + 1$
- conditions to detect overflow
  - $x_1[n], x_2[n] > 0$ and carry from (MSB)-1 to MSB occurs
  - $x_1[n], x_2[n] < 0$ and carry from (MSB)-1 to MSB does not occur
- no overflow if $x_1[n]$ and $x_2[n]$:
  - have opposite signs
  - lie in the shaded region (can scale prior to addition)



Overflow Avoidance Using Scaling

		MSB		MSB - 1			
$x_1[n]$		1	.	0	1	0	-0.75
$x_2[n]$		1	.	0	0	1	-0.875
$y[n]$	1	0	.	0	1	1	0.375

With Overflow

		MSB		MSB - 1				
$0.5x_1[n]$		1	.	1	0	1	0	-0.375
$0.5x_2[n]$		1	.	1	0	0	1	-0.4375
$0.5y[n]$	1	1	.	0	0	1	1	-0.8125

No Overflow with Scaling

- $x_1[n] = -0.75$, $x_2[n] = -0.875$, $y[n] = -1.625 \rightarrow$ overflow
- last 4 bits of $y[n]$ results in 0.375
- scale down $x_1[n]$ and $x_2[n]$ by a factor of 2
  - sign extension needed
  - presence of carry from MSB-1 to MSB ensures result is negative and > than -1

Series Addition Property

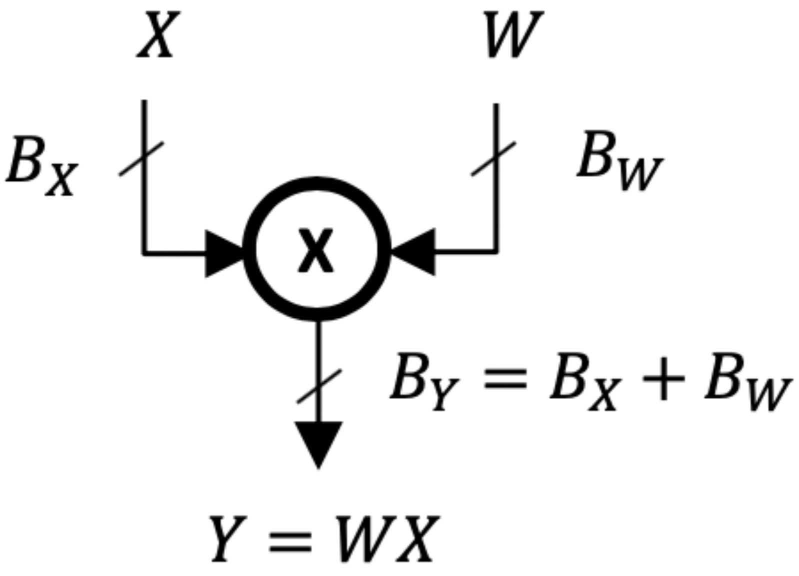
	0	.	0	1	0	1	0.3125	
	0	.	1	1	0	0	0.75	
	1	.	0	0	0	1	−0.9375	← Overflow
	1	.	1	0	0	0	−0.5	
1	0	.	1	0	0	1	0.5625	

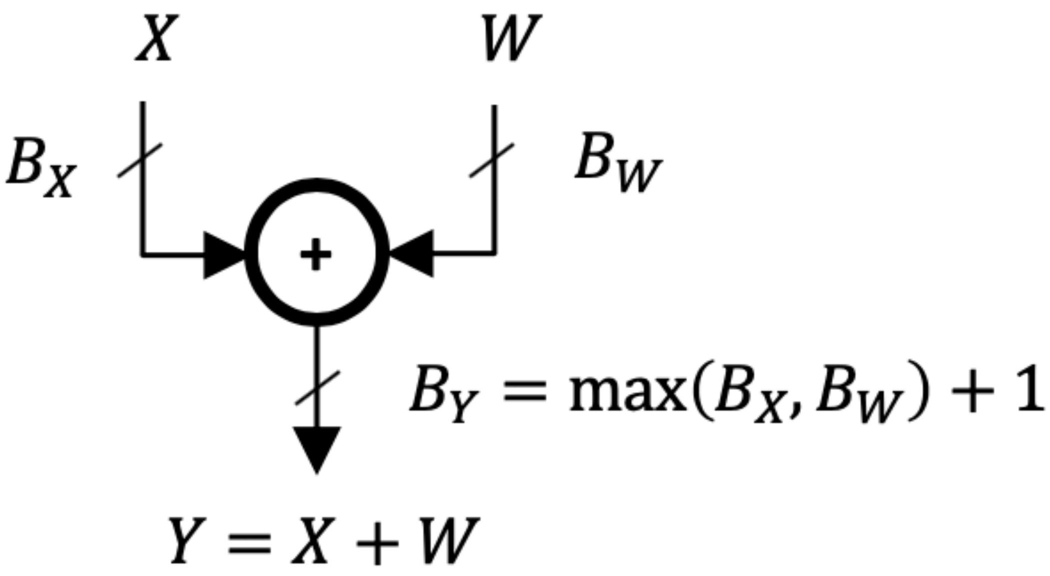
- in series of additions and subtractions, intermediate overflows are permitted as long as $-1 \leq y[n] < 1$
- example
  - addition of first 2 numbers results in overflow – allow it
  - final result is in correct range

Multiplication

- assume: $y[n] = x_1[n]x_2[n]$
- no overflow in multiplication ($|x_1[n]|$ and $|x_2[n]|$<1)
- exception: $x_1[n] = x_2[n] = -1$ since 1 has no 2's complement representation
- only source of quantization error is round-off
- to avoid round-off set → $B_y = B_{x_1} + B_{x_2}$
- other number representations
  - signed-magnitude representation

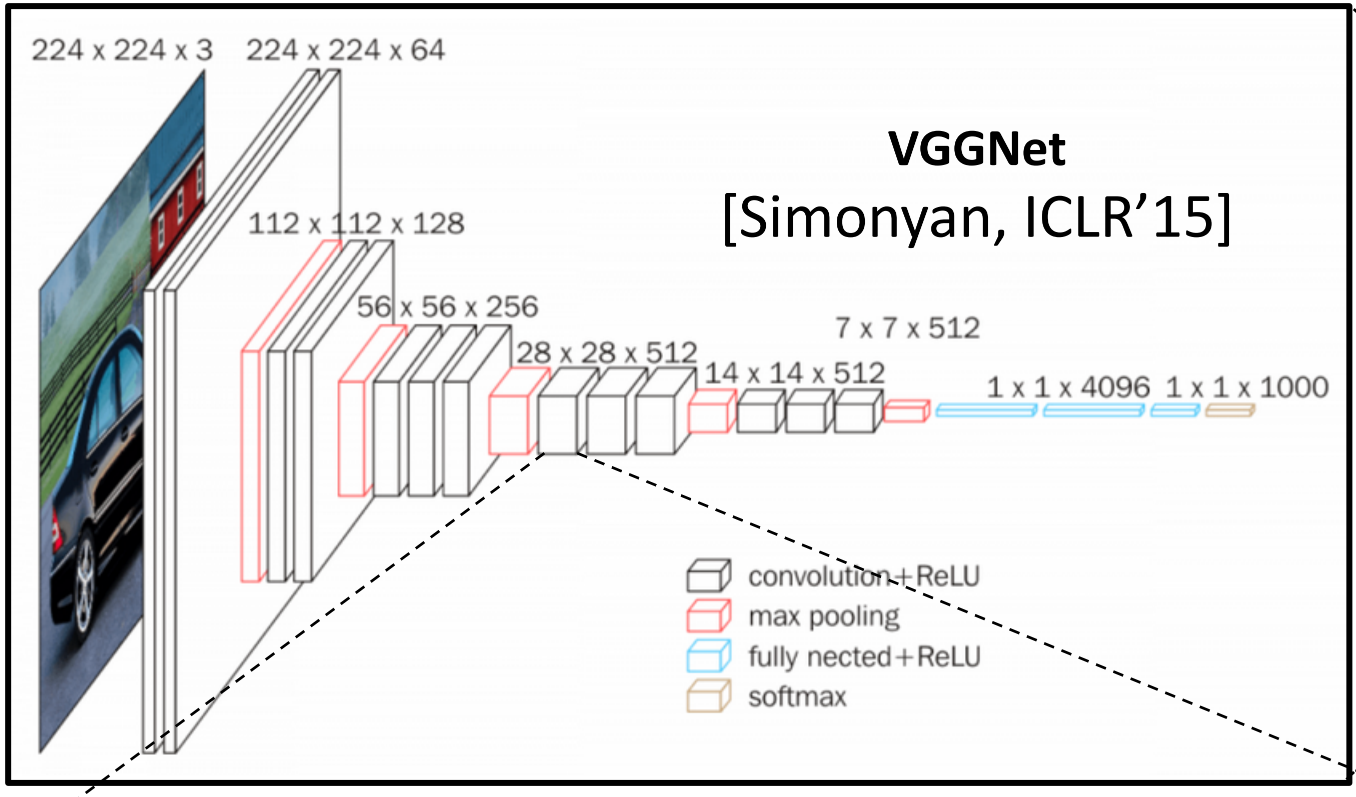
Precision of Multiply and Adds



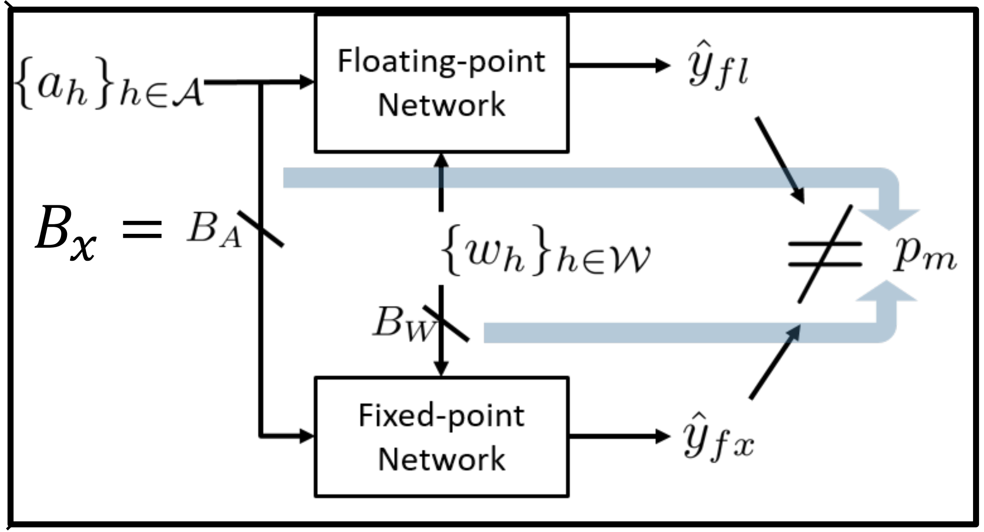


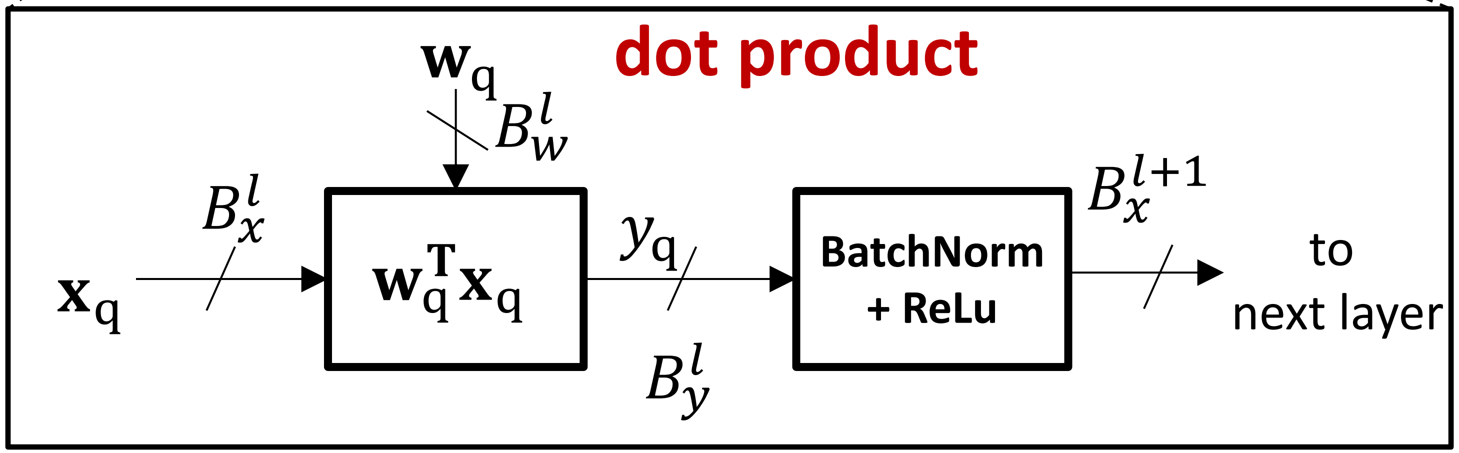
- bit-growth from input to output
- round-off or truncation to control bit growth

Fixed-point Dot Product



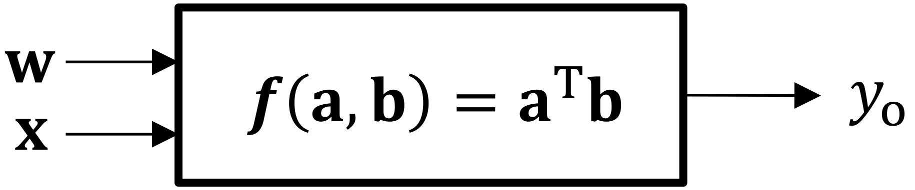
network





- what are the minimum values of B_x^l, B_w^l, and B_y^l $\forall\, l$ such that the network accuracy is within a Δ of floating-point network accuracy?

Floating-Point Dot Product

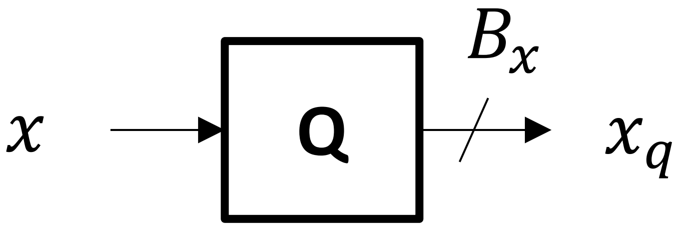


$$y_o = \sum_{j=0}^{N-1} w_j x_j$$

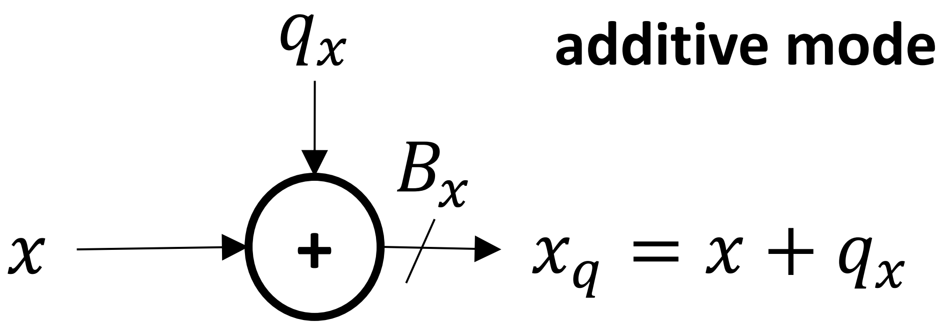
- represents an ideal for fixed-point implementations

Recall - Quantization Noise Model

quantizer symbol



additive model



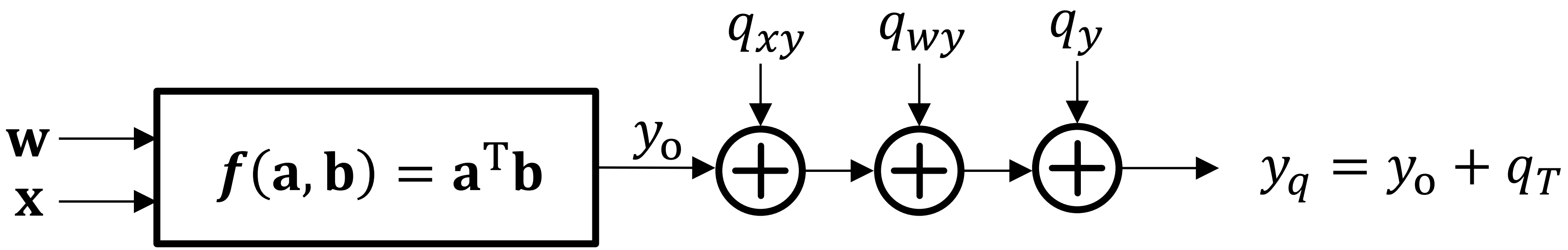
- additive model assumption: q_x is independent of x
- $SQNR$: signal-to-quantization noise ratio → accuracy measure
- ζ : peak-to-average (power) ratio → measure of 'peakiness' of signal distribution

$$\boxed{SQNR_x = 10\log_{10}\left[\frac{\sigma_x^2}{\sigma_{q_x}^2}\right]}$$

$$SQNR_x(dB) = 6B_x + 4.78 - \zeta_x\ (dB)$$

$$\zeta_x = \frac{x_m}{\sigma_x}$$

Fixed-point Dot Product

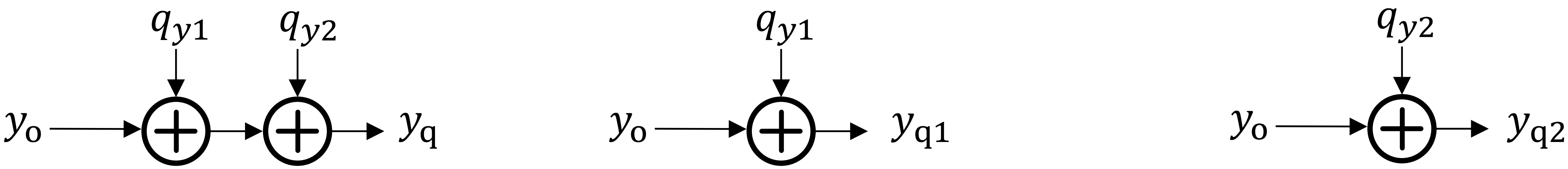


- three noise contributions need to be captured:

$$\sigma_{q_T}^2 = \sigma_{q_{xy}}^2 + \sigma_{q_{wy}}^2 + \sigma_{q_y}^2 = \sigma_{q_{iy}}^2 + \sigma_{q_y}^2$$

(input) (weights) (output)

SQNR Formula for Uncorrelated Additive Noise



$$\boxed{SQNR_{\mathrm{T}} = \left[\frac{1}{SQNR_{q_{y1}}} + \frac{1}{SQNR_{q_{y2}}}\right]^{-1}}$$ ("parallel" combination)

- $SQNR_T = \frac{\sigma_{y_o}^2}{\sigma_{q_T}^2}$: total SQNR; $q_T = q_{y1} + q_{y2}$ (total noise)
- $SQNR_{q_{y1}} = \frac{\sigma_{y_o}^2}{\sigma_{q_{y1}}^2}$: SQNR with only q_{y1} ; $SQNR_{q_{y2}} = \frac{\sigma_{y_o}^2}{\sigma_{q_{y2}}^2}$: SQNR with only q_{y2};

Ensuring a Dominant Noise Source

- Maximizing $SQNR_T$ by, e.g., ensuring that weight quantization noise is the limiting factor or analog noise is IMCs is the limiting factor

- $SQNR_{q_{y2}} = SQNR_{q_{y1}} + \alpha$ (dB) then $SQNR_T = SQNR_{q_{y1}} - 10\log_{10}(1 + 10^{-\frac{\alpha}{10}})$
- $SQNR_T = SQNR_{q_{y1}} - 0.5\text{ dB} \rightarrow \alpha = 9\text{ dB}$
- $SQNR_T = SQNR_{q_{y1}} - 1\text{dB} \rightarrow \alpha = 5.9\text{ dB}$
- $SQNR_T = SQNR_{q_{y1}} - 2\text{dB} \rightarrow \alpha = 2.3\text{ dB}$
- $SQNR_T = SQNR_{q_{y1}} - 3\text{dB} \rightarrow \alpha = 0\text{ dB}$ (0.5 LSB loss)

Two Approaches to Weight Quantization

1) perturbation model

$$w_q = w + \Delta w$$

2) additive noise model

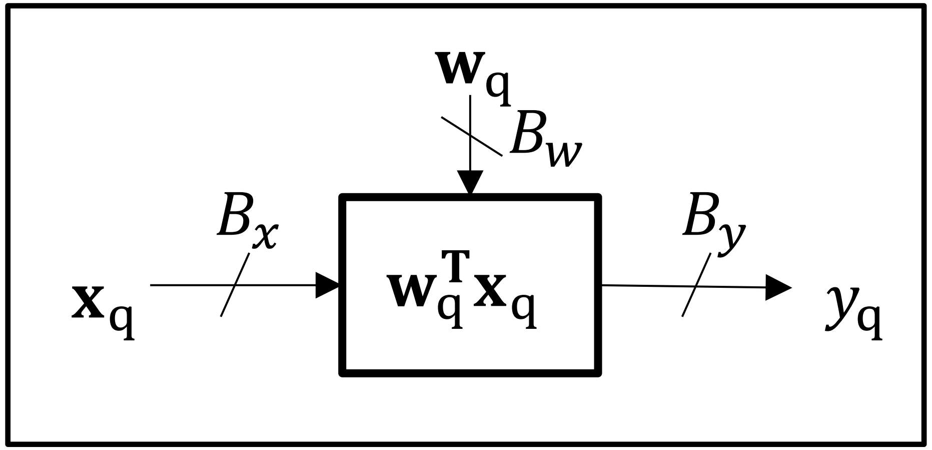
$$w_q = w + q_w$$

- How to model weights?

1) weights as deterministic variables → weight quantization as a perturbation (fixed coefficients, e.g., FIR filters) – **perturbation model**
2) weights as random variables (RVs) → weight quantization as statistical noise (weight ensemble, e.g., DNNs) – **noise model**

Fixed-point Dot Product – perturbation model

Fixed-Point Dot Product



- floating-point output (ideal): $y_o = \sum_i x_i h_i$
- fixed-point output:

$$y_q = Q\left[\sum_i (x_i + q_{x_i})(h_i + \Delta h_i)\right] = Q\left[y_o + \sum_i (x_i \Delta h_i + h_i q_{x_i} + q_{x_i}\Delta h_i)\right]$$

$$= y_o + \sum_i (x_i \Delta h_i + h_i q_{x_i} + \underbrace{q_{x_i}\Delta h_i}_{\text{ignore}}) + q_y = y_o + q_T$$

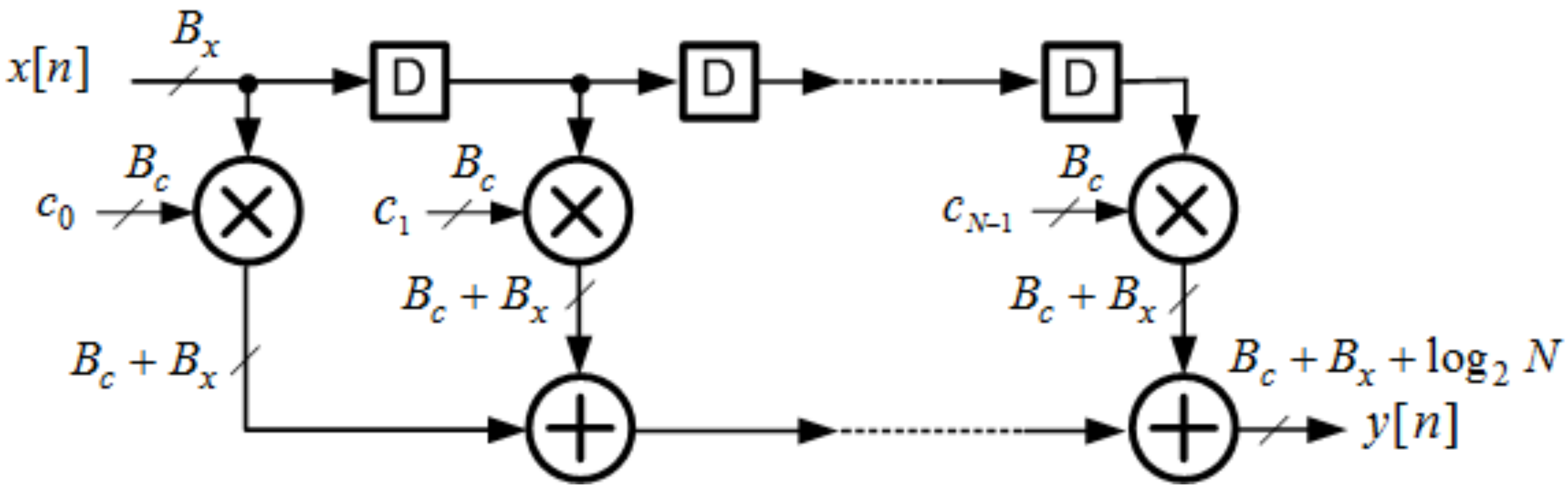
$$q_T = q_{hy}(= \sum x_i \Delta h_i) + q_{xy}(= \sum h_i q_{x_i}) + q_y$$

Total Output Quantization Noise

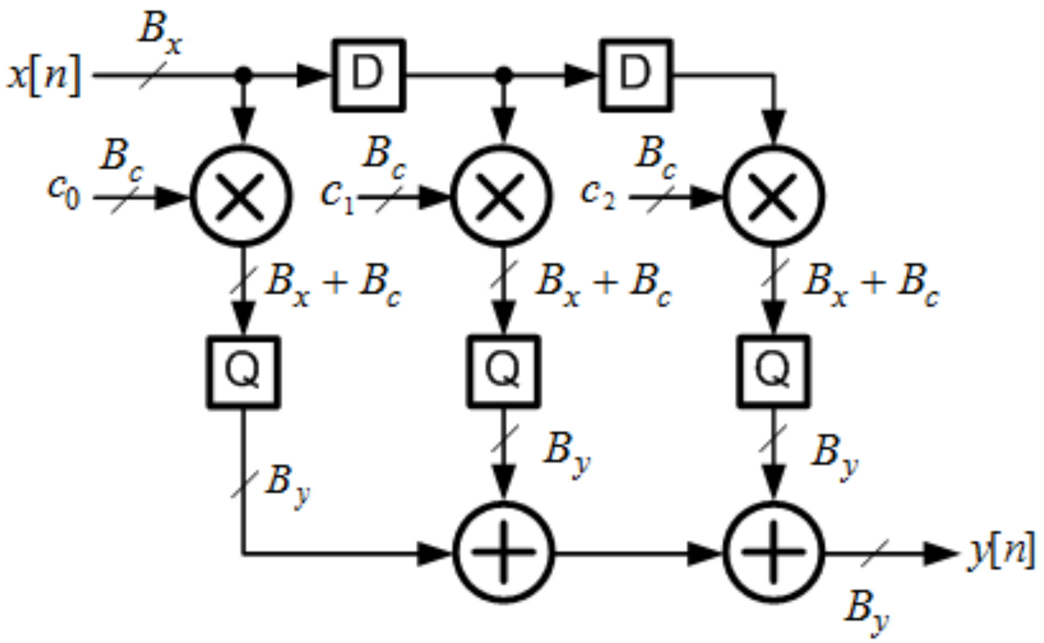
$$q_T = \underset{\substack{\uparrow\\\text{coefficient quantization}}}{\overline{q_{hy}}} + \underset{\substack{\uparrow\\\text{input quantization}}}{\overline{q_{xy}}} + \underset{\substack{\uparrow\\\text{output quantization}}}{\overline{q_y}} \rightarrow \sigma_{q_T}^2 = \sigma_{q_{hy}}^2 + \sigma_{q_{xy}}^2 + \sigma_{q_y}^2$$

- $\boldsymbol{R} = \mathbf{E}[XX^T]$: data covariance matrix
- For uncorrelated inputs: $\sigma_{q_{hy}}^2 = \Delta\boldsymbol{h}^T\boldsymbol{R}\Delta\boldsymbol{h} = \sigma_x^2 \sum_i \Delta h_i^2$; $\sigma_{y_o}^2 = \boldsymbol{h}^T\boldsymbol{R}\boldsymbol{h} = \sigma_x^2 \sum_i h_i^2$

Accumulator (Output) Quantization via Bit Growth Criterion (BGC)



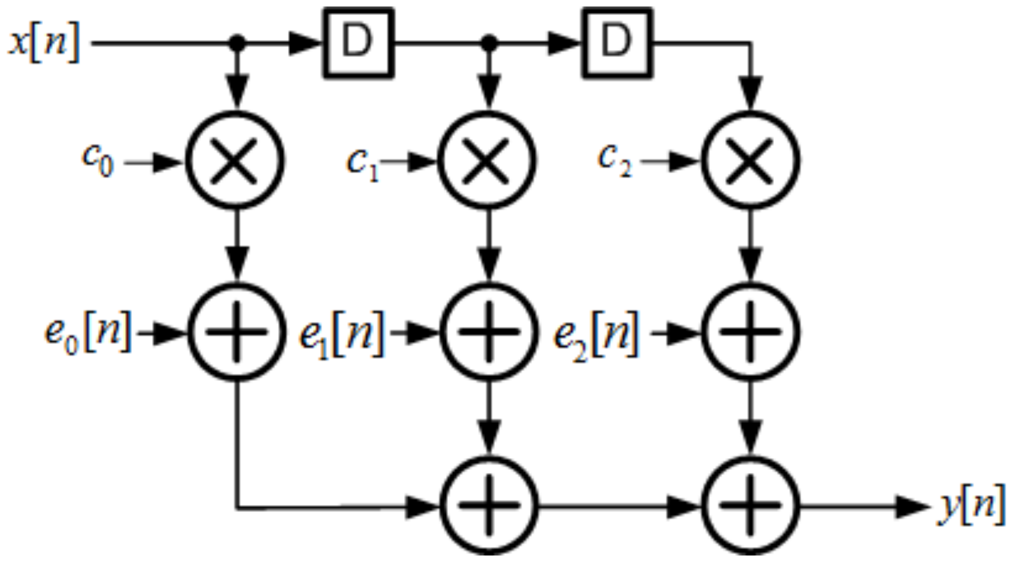
- commonly known as the 'bit growth' phenomenon
- conservative approach → maximum precision solution
- to avoid overflow completely $\log_2 N$ additional bits needed
- need $B_x \times B_c$ bit multipliers and $B_x + B_c + \log_2 N$ bit adders



Reduced Precision Accumulation

- given a target output SQNR ($SQNR_{q_y}$)
  - determine the peak value of $y[n]$ and its variance (via analysis)
  - Find B_y ($SQNR_{q_y} = 6B_y + 4.8 - PAR_y$) → this value of $B_y \leq B_x + B_c + \log_2 N$
  - round-off multiplier outputs to B_y bits
- use series addition property of 2's complement to permit overflow or allow for a non-zero clipping probability

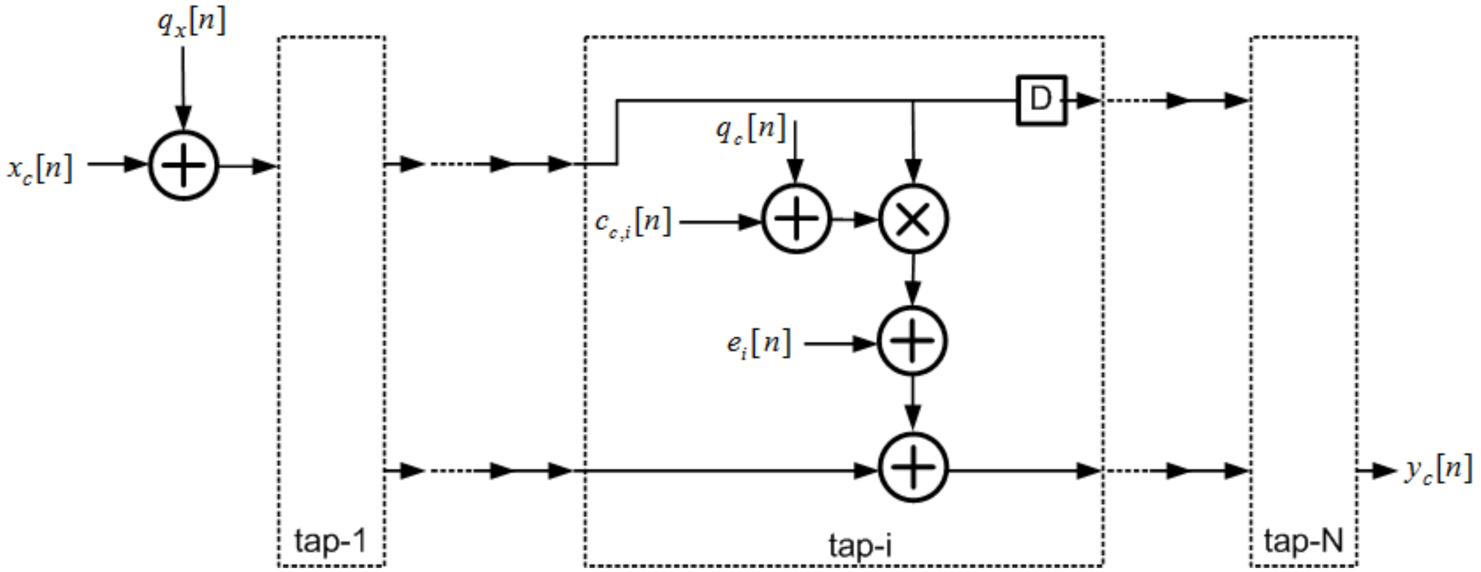
Accumulator Round-off Error Model



- Additive noise $e_i[n]$ of variance: $\sigma^2_{e,i} = \frac{2^{-2B_y}}{3}$

- Total round-off noise at output: $\sigma^2_{q_y} = N\frac{2^{-2B_y}}{3}$

Complete Finite-precision FIR Filter



- total quantization noise variance at output:

$$\sigma^2_{q_T} = \sigma^2_{q_{xy}} + \sigma^2_{q_{hy}} + \sigma^2_{q_y} = \sigma^2_{q_{iy}} + \sigma^2_{q_y}$$

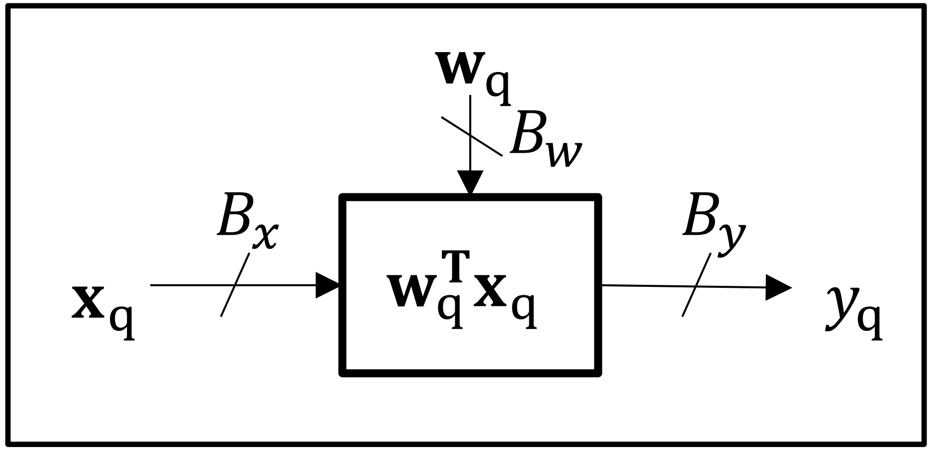
$$= \sigma^2_{q_x}\boldsymbol{h}^T\boldsymbol{h} + \Delta\boldsymbol{h}^T\boldsymbol{R}\Delta\boldsymbol{h} + N\frac{2^{-2B_y}}{3}$$

(input) (coeff.) (accumulator round-off/output quantization)

Total Output SQNR

$$SQNR_T = \frac{\sigma_{y_o}^2}{\sigma_{q_T}^2} = \frac{\sigma_{y_o}^2}{\sigma_{q_{xy}}^2 + \sigma_{q_{hy}}^2 + \sigma_{q_y}^2} = \frac{\sigma_{y_o}^2}{\sigma_{q_{xy}}^2 + \sigma_{q_{hy}}^2 + \sigma_{q_y}^2}$$

$$= \frac{\boldsymbol{h}^T \boldsymbol{R} \boldsymbol{h}}{\underbrace{\Delta \boldsymbol{h}^T \boldsymbol{R} \Delta \boldsymbol{h}}_{\text{coefficient quantization noise}} + \underbrace{\sigma_{q_x}^2 \boldsymbol{h}^T \boldsymbol{h}}_{\text{input quantization noise}} + \underbrace{\sigma_{q_y}^2}_{\text{output quantization noise}}}$$

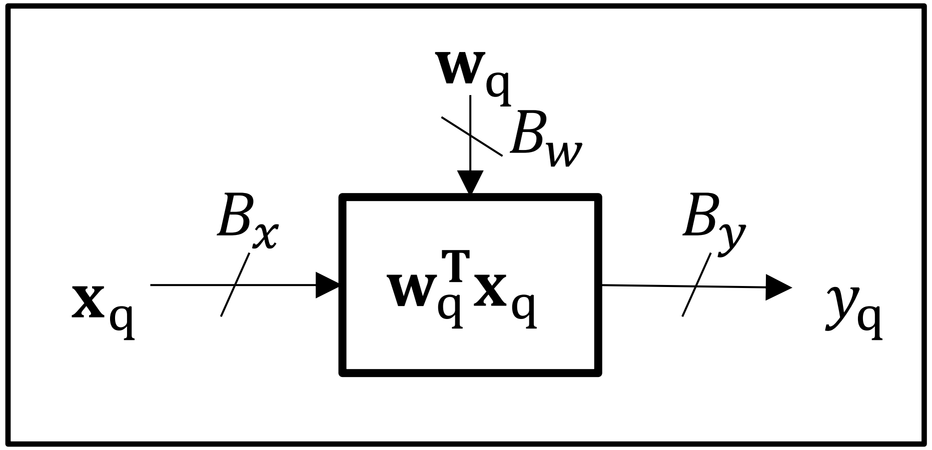


SQNR Formula

$$\boxed{SQNR_\mathrm{T} = \left[\frac{1}{SQNR_{q_{iy}}} + \frac{1}{SQNR_{q_y}}\right]^{-1}}$$

Limited by $SQNR_{q_{iy}}$

- $SQNR_T = \frac{\sigma_{y_o}^2}{\sigma_{q_T}^2}$: total SQNR; $SQNR_{q_{iy}} = \frac{\sigma_{y_o}^2}{\sigma_{q_{iy}}^2}$: SQNR with only q_{iy}
- $SQNR_{q_y} = \frac{\sigma_{y_o}^2}{\sigma_{q_y}^2}$: SQNR with only q_y (output quantization);



$$\boxed{SQNR_T = \left[\frac{1}{SQNR_{q_{iy}}} + \frac{1}{SQNR_{q_y}}\right]^{-1}} \longrightarrow \boxed{\text{Limited by } SQNR_{q_{iy}}}$$

- Choose $SQNR_{q_y}(dB) \geq SQNR_{q_{iy}}(dB) + 9$ to minimize ($< 0.5dB$) its impact on $SQNR_T$

Example: Fixed-Point Dot Product

- Given:
  - floating point coefficient vector: $\boldsymbol{h}_{fl} = [-0.3333, 0.5555, -0.3333]$
  - input $x_c[n]$ uncorrelated & uniformly distributed between ± 1
  - $B_x = 7, B_h = 5, \boldsymbol{h}_q = [-0.3125 \quad 0.5625 \quad -0.3125]$

- Full bit-growth $\rightarrow B_y = 7 + 5 + \log_2 3 = 14$
- Calculate $SQNR_T$ for $B_y = 10, 5$?
- $\sigma_x^2 = \frac{\Delta^2}{12} = \frac{4}{12} = \frac{1}{3} \rightarrow PAR_x = 10\log_{10}\left[\frac{(1)^2}{\sigma_x^2}\right] = 4.77$ dB
- $SQNR_x = 42 + 4.8 - 4.8 = 42dB = 10\log_{10}\frac{\sigma_x^2}{\sigma_{q_x}^2}$
- $\sigma_{q_x}^{\ 2} = \frac{1}{12\times 2^{14-2}} = 2.0345\times 10^{-5} \approx \frac{1}{3\times 10^{4.2}}$

$$\sigma_{q_y}^2 = \sigma_{q_{iy}}^2 + \sigma_{q_y}^2 = \sigma_{q_{xy}}^2 + \sigma_{q_{hy}}^2 + \sigma_{q_y}^2$$

$$= \sigma_{q_x}^2 \boldsymbol{h}^T\boldsymbol{h} + \Delta\boldsymbol{h}^T\boldsymbol{R}\Delta\boldsymbol{h} + N\frac{2^{-2B_y}}{3}$$

(1.0798×10^{-5}) (3.0476×10^{-4})

(input) (coeff.) (output)

$$SQNR_{q_{iy}} = 10\log_{10}\left[\frac{\sigma_{y_o}^2}{\sigma_{q_{iy}}^2}\right] = 10\log_{10}\left[\frac{0.1769}{1.0798\times10^{-5}+3.0476\times10^{-4}}\right] = 27.48 \text{ dB}$$

$SQNR_T \leq SQNR_{q_{iy}}$ (upper bound)

$$\sigma_{q_T}^2 = \sigma_{q_{iy}}^2 + \sigma_{q_y}^2 = \sigma_{q_{xy}}^2 + \sigma_{q_{hy}}^2 + \sigma_{q_y}^2$$

$$= \sigma_{q_x}^2 \boldsymbol{h}^T \boldsymbol{h} + \Delta \boldsymbol{h}^T \boldsymbol{R} \Delta \boldsymbol{h} + N \frac{2^{-2B_y}}{3}$$

(1.0798×10^{-5}) (3.0476×10^{-4})

(input) (coeff.) (output)

- $B_y = 10 \rightarrow \sigma_{q_y}^2 = 3\frac{2^{-2B_y}}{3} \approx 10^{-6} \rightarrow\ SQNR_{q_y} = 10\log_{10}\left[\frac{0.1769}{10^{-6}}\right] = 52\text{ dB}$ (same as 'full bit growth' $\rightarrow$ $B_y = 14$)

  $$SQNR_T = 10\log_{10}\left[\frac{\sigma_y^2}{\sigma_{q_T}^2}\right] = 10\log_{10}\left[\frac{0.1769}{1.0798\times10^{-5}+3.0476\times10^{-4}+10^{-6}}\right] = 27.4727\text{ dB}$$

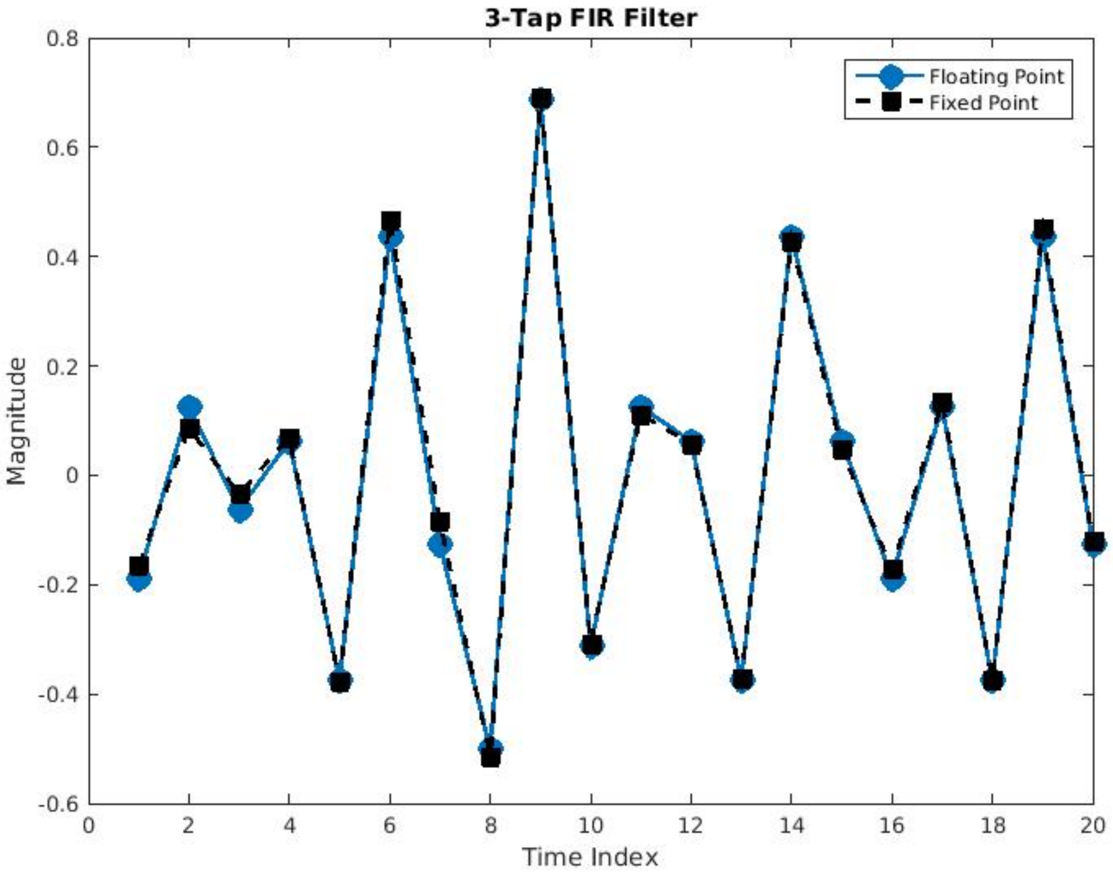
- $B_y = 5 \rightarrow \sigma_{q_y}^2 = 3\times\frac{2^{-2B_y}}{3} \approx 9.7656\times10^{-4} \rightarrow SQNR_{q_y} = 22.58\text{ dB}$

  $$SQNR_T = 10\log_{10}\left[\frac{0.1769}{1.0798\times10^{-5}+3.0476\times10^{-4}+9.7656\times10^{-4}}\right] = 21.3647\text{ dB}$$ (~ 1 LSB (6 dB) loss)

- $B_y = 6 \rightarrow SQNR_{q_y} = 28.6\text{ dB} \rightarrow SQNR_T = 24.99\text{ dB}$ (< 0.5 LSB (3 dB) loss)

Time-domain Plot for $B_y = 5$

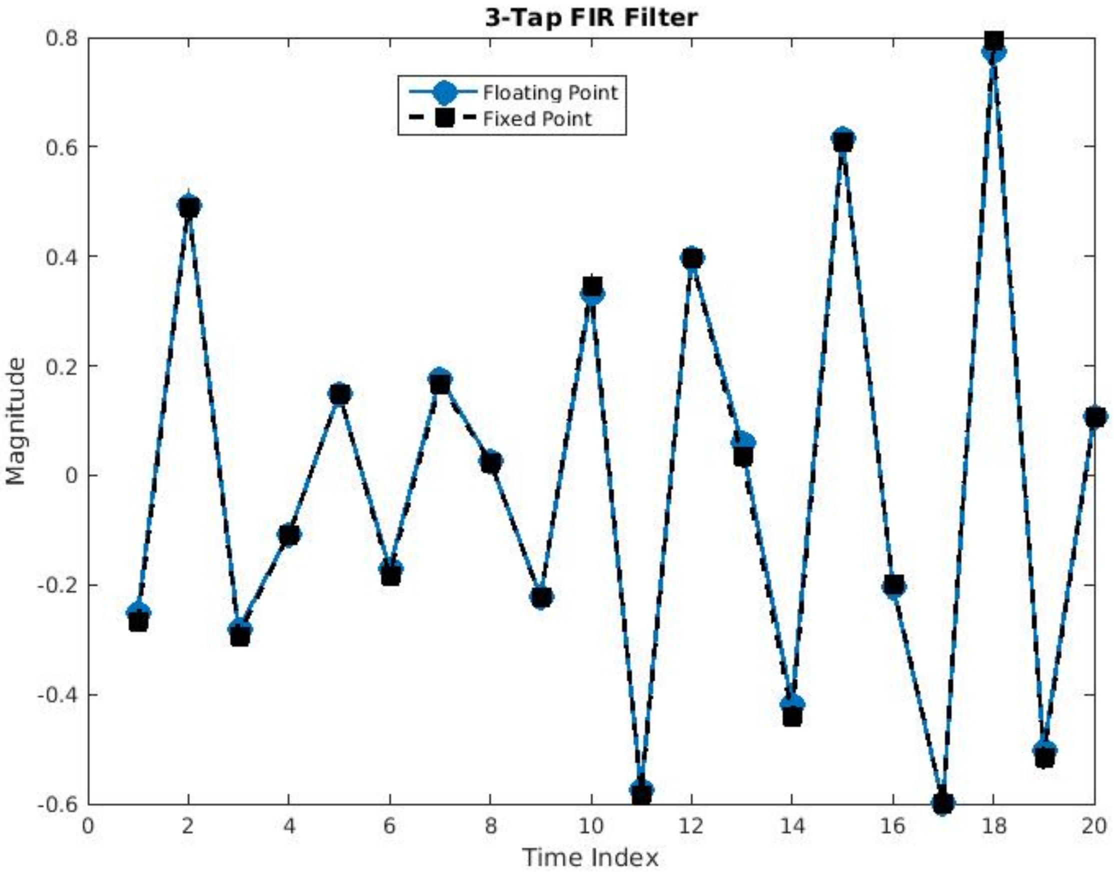
evaluated (analysis): $SQNR_T = 21.3647$ dB
estimated (sim): $SQNR_T = 21.4372$ dB



Time-domain Plot for

$B_y = 10$

evaluated (analysis): $SQNR_T = 27.4739$ dB
estimated (sim): $SQNR_T = 27.4603$ dB



Fixed-point Dot Product – noise model

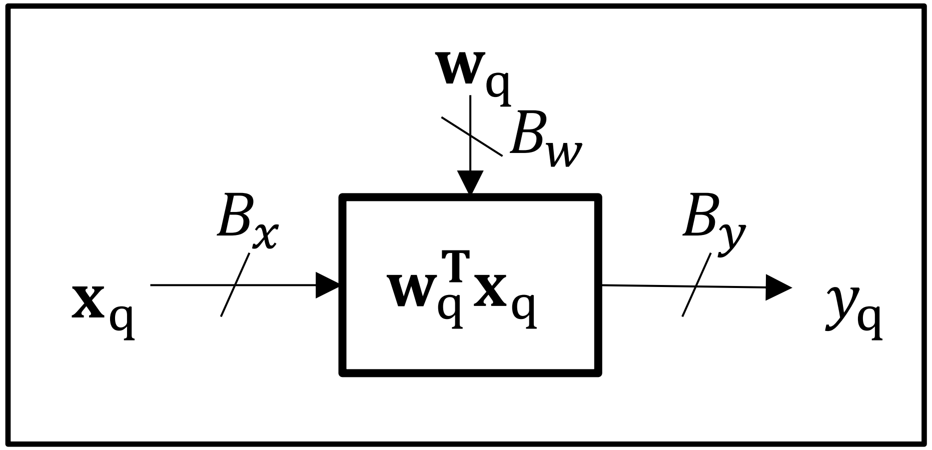
Fundamental Limits on the Precision of In-memory Architectures

(Invited Talk)

Sujan K. Gonugondla, Charbel Sakr, Hassan Dbouk, and Naresh R. Shanbhag
(gonugon2,sakr2,hdbouk2,shanbhag)@illinois.edu
Department of Electrical and Computer Engineering
University of Illinois at Urbana-Champaign

2020 IEEE International Conference on Computer-Aided Design (ICCAD)
November 2-5, 2020.

Fixed-Point Dot Product



- floating-point output (ideal):

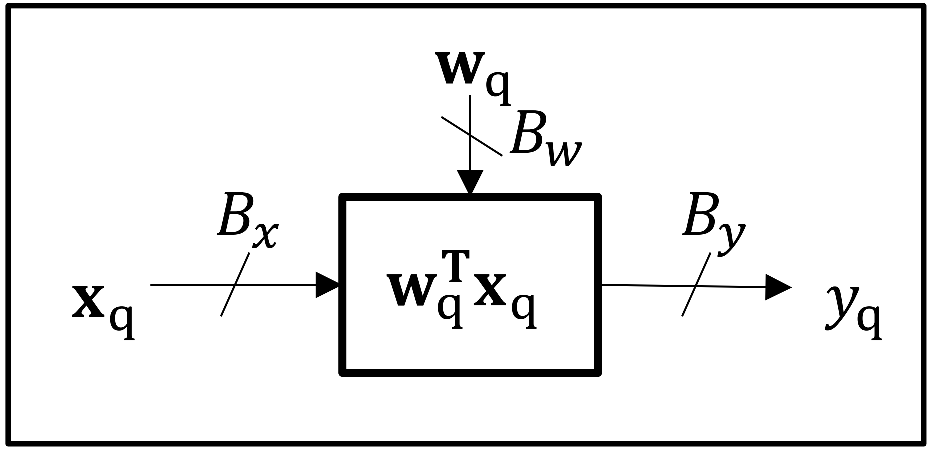
$$y_o = \sum_i w_i x_i = \mathbf{w}^\mathrm{T}\mathbf{x}$$

- fixed-point output:

$$y_q = Q[\mathbf{w}^\mathrm{T}\mathbf{x}] = (\mathbf{w} + \mathbf{q}_w)^\mathrm{T}(\mathbf{x} + \mathbf{q}_x) + q_y \approx \mathbf{w}^\mathrm{T}\mathbf{x} + \mathbf{w}^\mathrm{T}\mathbf{q}_x + \mathbf{q}_w^\mathrm{T}\mathbf{x} + q_y$$

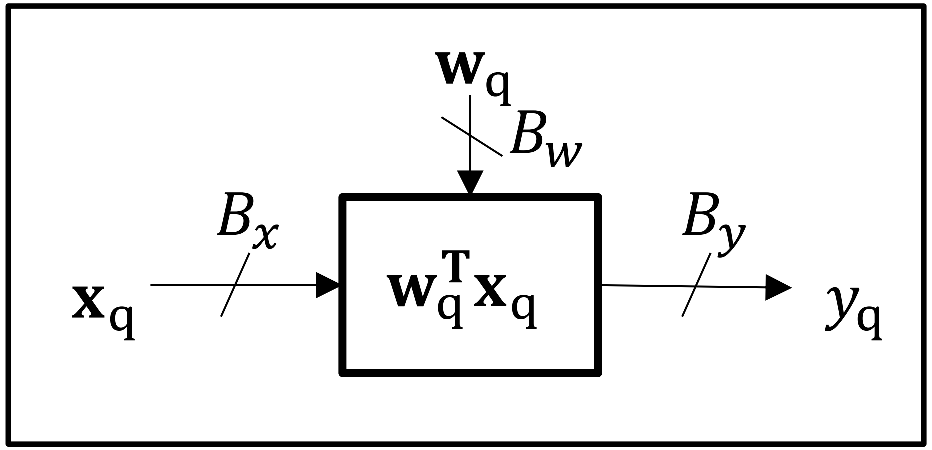
$$= y_o + q_{xy} + q_{wy} + q_y = y_o + q_{iy} + q_y = y_o + q_T$$

$$q_T = q_{wy}(= \mathbf{q}_w^\mathrm{T}\mathbf{x}) + q_{xy}(= \mathbf{w}^\mathrm{T}\mathbf{q}_x) + q_y$$



$$y_q = \overbrace{\mathbf{w}^{\mathrm{T}}\mathbf{x}}^{\overbrace{y_o}^{\text{ideal FL output}}} + \underbrace{q_{iy}}_{\text{input quantization noise } \textit{reflected} \text{ at the output}} + \overbrace{q_y}^{\text{output quantization noise}}$$

- $\sigma_{y_o}^2 = N\sigma_w^2 E[x^2]$; $\sigma_{q_y}^2 = \frac{\Delta_y^2}{12}$; $\sigma_{q_{iy}}^2 = \frac{N}{12}(\Delta_w^2 E[x^2] + \Delta_x^2 \sigma_w^2)$
- note: weights and weight quantization is modeled as RVs

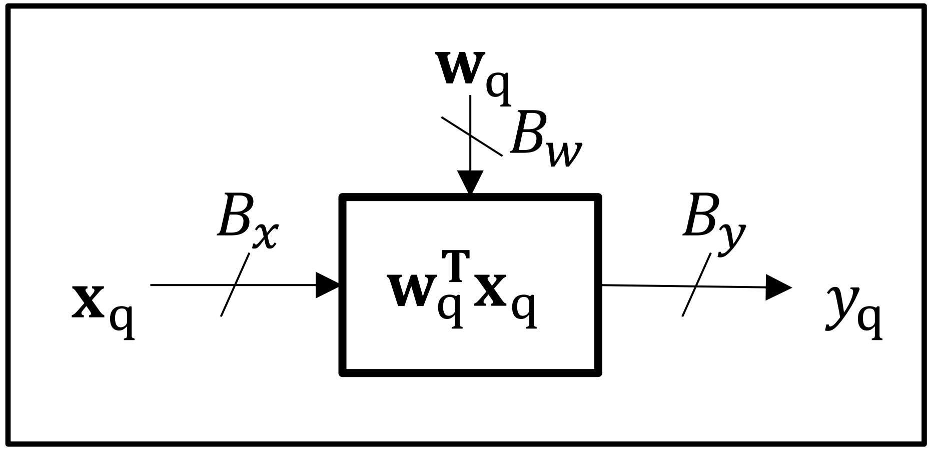


SQNR Formula

$$SQNR_\mathrm{T} = \left[\frac{1}{SQNR_{q_{iy}}} + \frac{1}{SQNR_{q_y}}\right]^{-1}$$

Limited by $SQNR_{q_{iy}}$

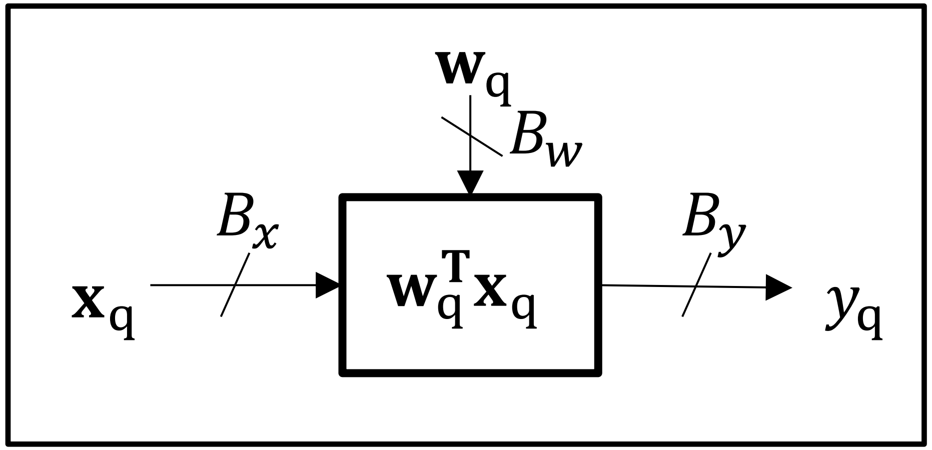
- $SQNR_T = \dfrac{\sigma_{y_o}^2}{\sigma_{q_T}^2}$: total SQNR; $SQNR_{q_{iy}} = \dfrac{\sigma_{y_o}^2}{\sigma_{q_{iy}}^2}$: SQNR with only q_{iy}
- $SQNR_{q_y} = \dfrac{\sigma_{y_o}^2}{\sigma_{q_y}^2}$: SQNR with only q_y (output quantization);



$$SQNR_\mathrm{T} = \left[\frac{1}{SQNR_{q_{iy}}} + \frac{1}{SQNR_{q_y}}\right]^{-1}$$

Limited by $SQNR_{q_{iy}}$

- Choose $SQNR_{q_y}(dB) \geq SQNR_{q_{iy}}(dB) + 9$ to minimize ($< 0.5dB$) its impact on $SQNR_\mathrm{T}$



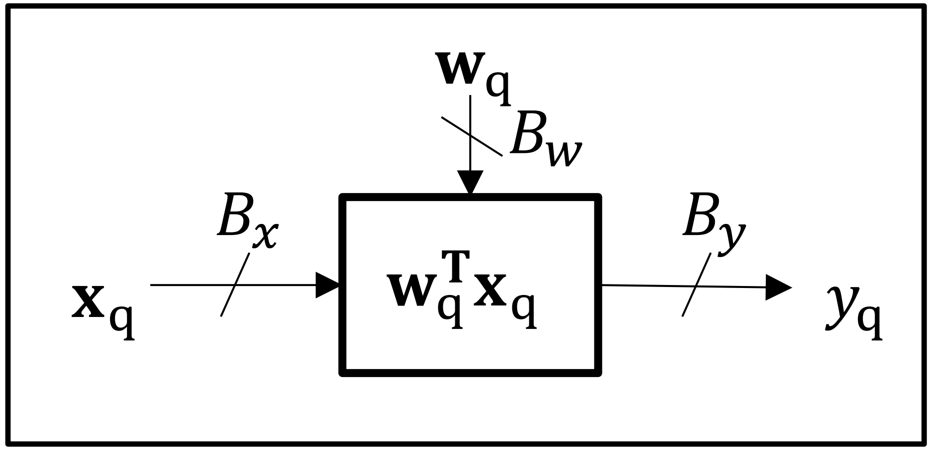
$SQNR_{q_{iy}}$

(SQNR due to input quantization)

$$SQNR_{q_{iy}}(dB) = \frac{\sigma_{y_o}^2}{\sigma_{q_{iy}}^2}$$

$$= 6(B_x + B_w) + 4.8 - [\zeta_x(dB) + \zeta_w(dB)] - 10\log_{10}\left(\frac{2^{2B_x}}{\zeta_x} + \frac{2^{2B_w}}{\zeta_w}\right)$$

- assumes $B_y \rightarrow \infty$ (no output quantization)
- establishes an upper bound on the total SQNR



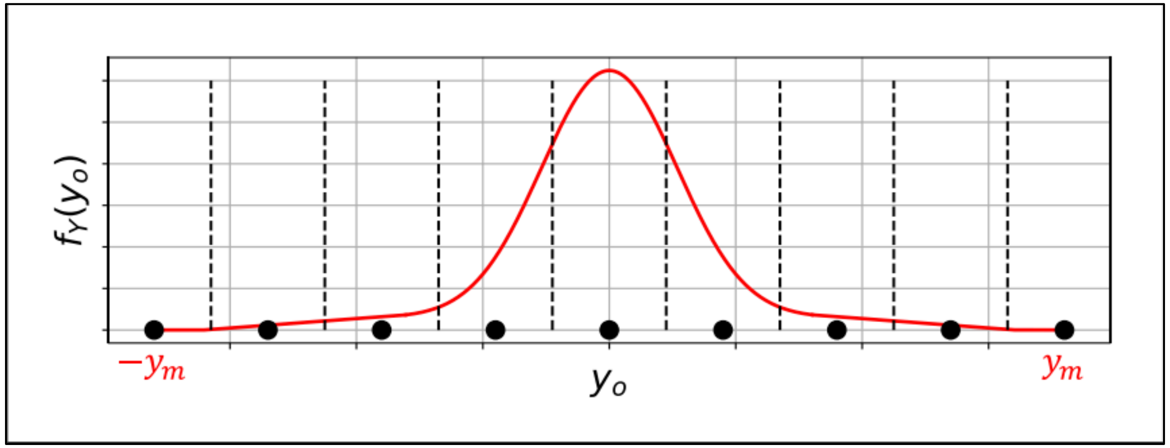
$SQNR_{q_y}$

(SQNR due to output quantization)

$$SQNR_{q_y}(dB) = 6B_y + 4.8 - [\zeta_x(dB) + \zeta_w(dB)] - 10\log_{10}(N)$$

- assumes $B_x, B_w \rightarrow \infty$ (no input quantization)
- But for fixed B_y: $SQNR_{q_y}(dB)$ **reduces with N** (N in hundreds in DNNs) $\rightarrow$ increase B_y
- But large B_y $\rightarrow$ leads to very large accumulator bit widths
- **How to choose output precision B_y?**

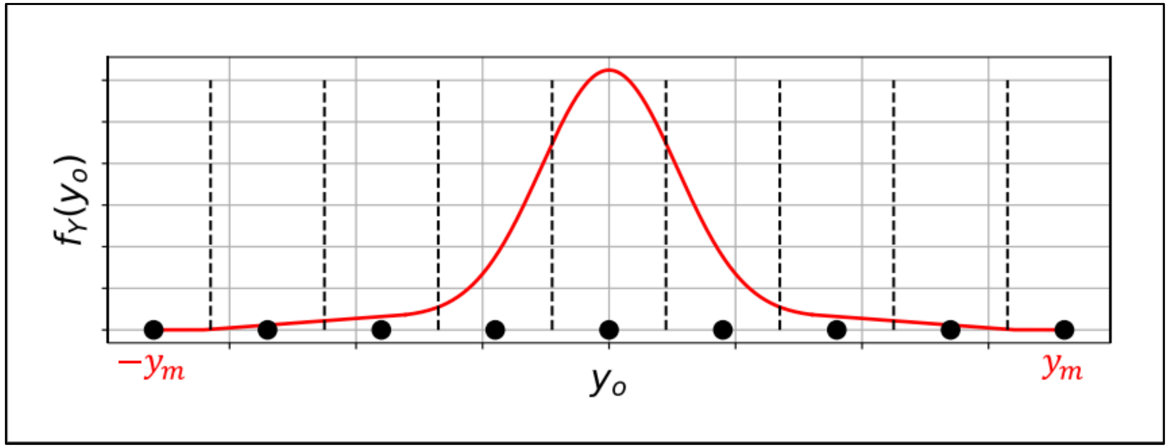
Choosing Output Precision B_y



$$y_q = \mathbf{w}^{\mathrm{T}}\mathbf{x} + q_y$$

- B_y is the accumulator precision in digital architectures → accumulator complexity dominates power in low-precision DNNs
  - e.g., 32b accumulator 10× more power than a 3×1-b multiplier in 28nm CMOS – hence research on low-resolution accumulation [Sakr ICLR19; Wang NeurIPS'18]
- B_y is the ADC precision in in-memory architectures → ADCs can dominate (~80%) latency and power when implementing DNNs [Kim ISLPED'18, Rekhi DAC'20]

Bit Growth Criterion (BGC) for Choosing B_y

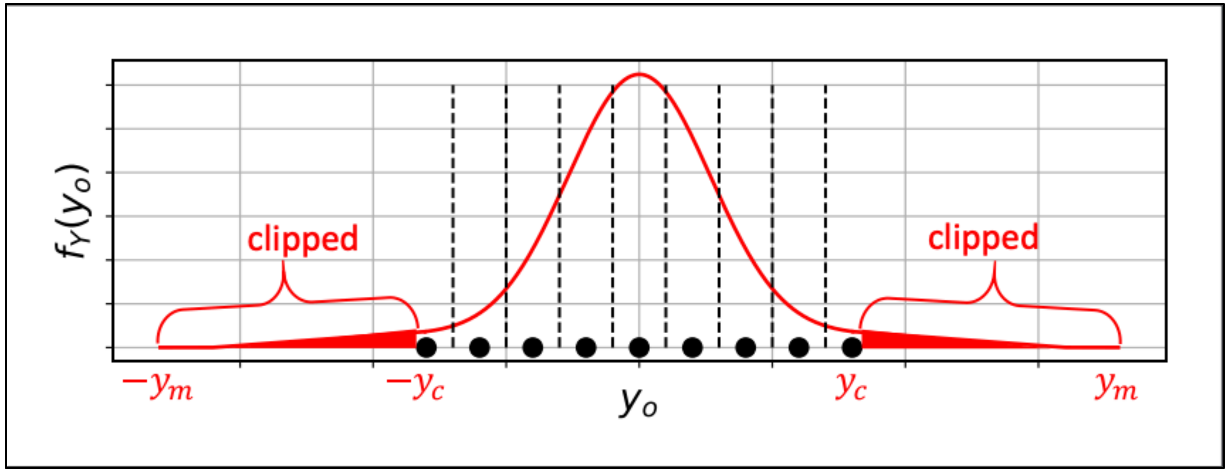


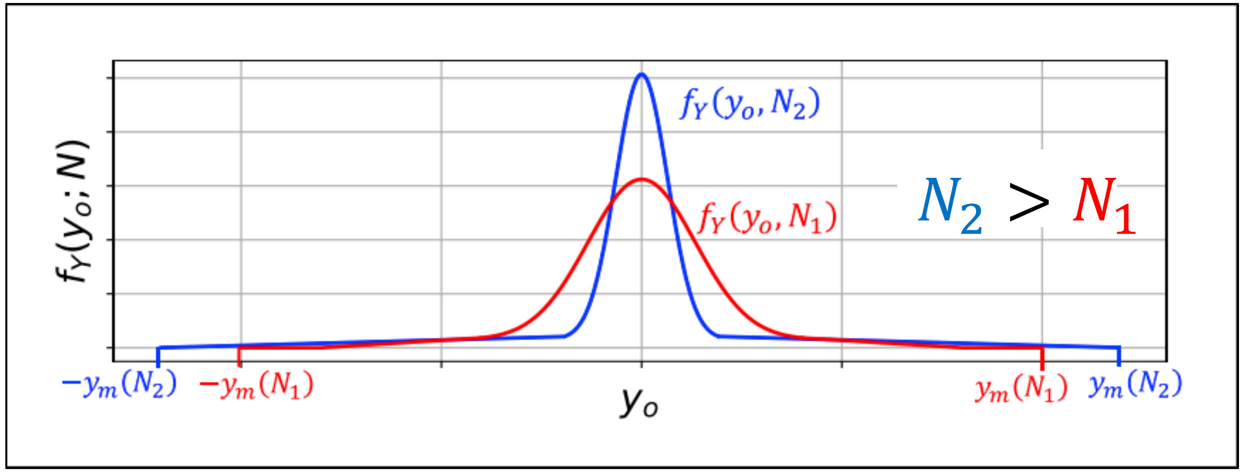
$$\boxed{B_y = B_x + B_w + \log_2(N)}$$

$$\boxed{SQNR_{q_y}^{BGC}(dB) = 6(B_x + B_w) + 4.8 - [\zeta_x(dB) + \zeta_w(dB)] + 10\log_{10}(N)}$$

- commonly employed in digital architectures and network design
- B_y (accumulator precision) and $SQNR_{q_y}$ both **increase with N**

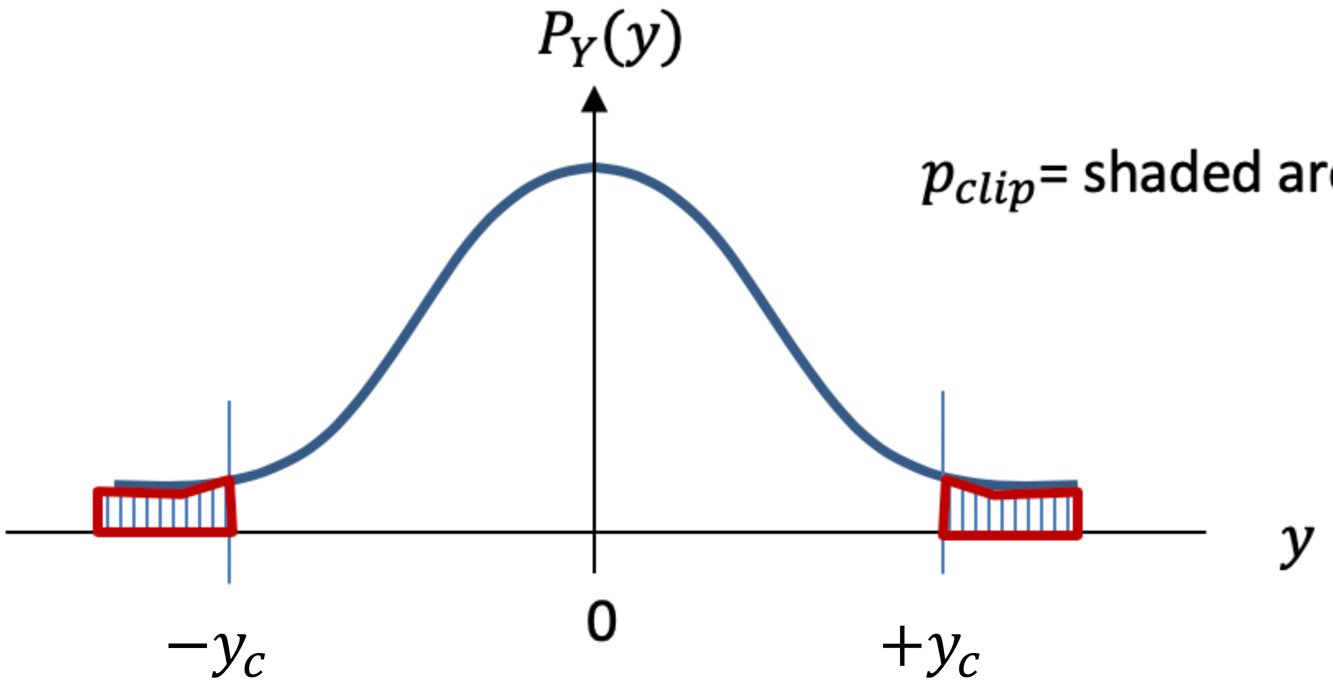
Proposed - Minimum Precision Criterion (MPC)





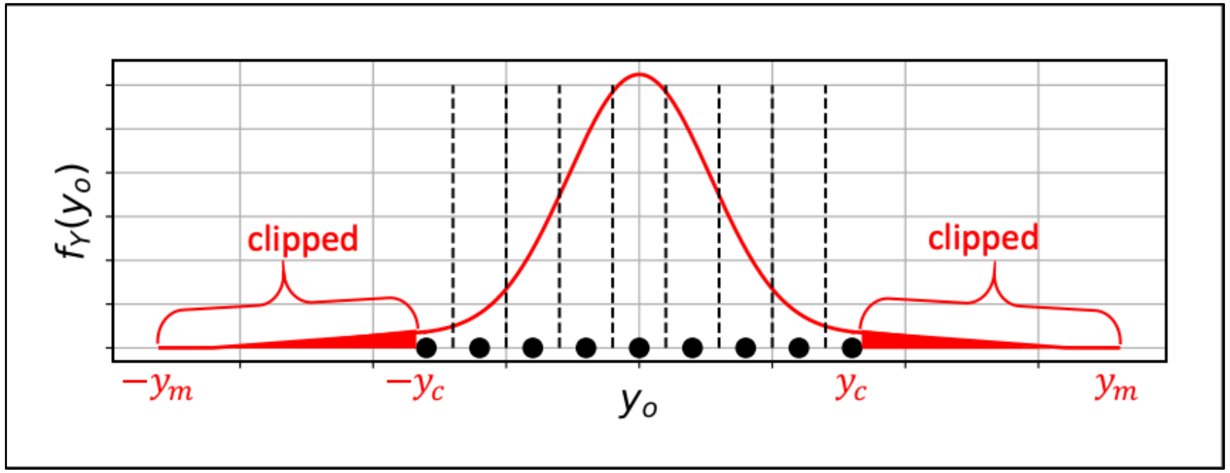
- allow for a non-zero but small probability of clipping (p_c) – BGC avoids clipping
- exploits reduction in $\frac{\sigma}{\mu}$ of y_o with N (Central Limit Theorem) to reduce PAR

Clipping Probability



- output can be clipped to $[-y_c, +y_c]$ to limit its range → reduces PAR_y → improves SQNR
- y_c (>0): clipping level
- $p_c = \Pr\{|y| > y_c\}$ - clipping probability $= 2Q\left(\zeta_y^{MPC}\right)$ if $y \sim \mathcal{N}(0, \sigma_y^2)$ (Gaussian)

$SQNR_{q_y}^{MPC}$



$$SQNR_{q_y}^{MPC}(dB) = 6B_y + 4.8 - \zeta_y^{MPC}(dB) - 10\log_{10}\left(1 + p_c\frac{\sigma_{cc}^2}{\sigma_{q_y}^2}\right)$$

- $\sigma_{cc}^2 = \mathrm{E}\left\{\left||y| - y_c\right|^2 \Big| |y| > y_c\right\}$;
- exhibits a trade-off between clipping noise and quantization noise → setting $y_c = 4\sigma_{y_o}$ offers the optimum trade-off

Summary - BGC, tBGC and MPC

- BGC

$$\boxed{SQNR_{q_y}^{BGC}(dB) = 6(B_x + B_w) + 4.8 - [\zeta_x(dB) + \zeta_w(dB)] + 10\log_{10}(N)}$$

$$\boxed{B_y^{BGC} = B_x + B_w + \log_2(N)}$$

- tBGC $\boxed{SQNR_{q_y}(dB) = 6B_y + 4.8 - [\zeta_x(dB) + \zeta_w(dB)] - 10\log_{10}(N)}$

- MPC $\boxed{SQNR_{q_y}^{MPC}(dB) = 6B_y + 4.8 - \zeta_y^{MPC}(dB) - 10\log_{10}\left(1 + p_c\frac{\sigma_{cc}^2}{\sigma_{q_y}^2}\right)}$

Example

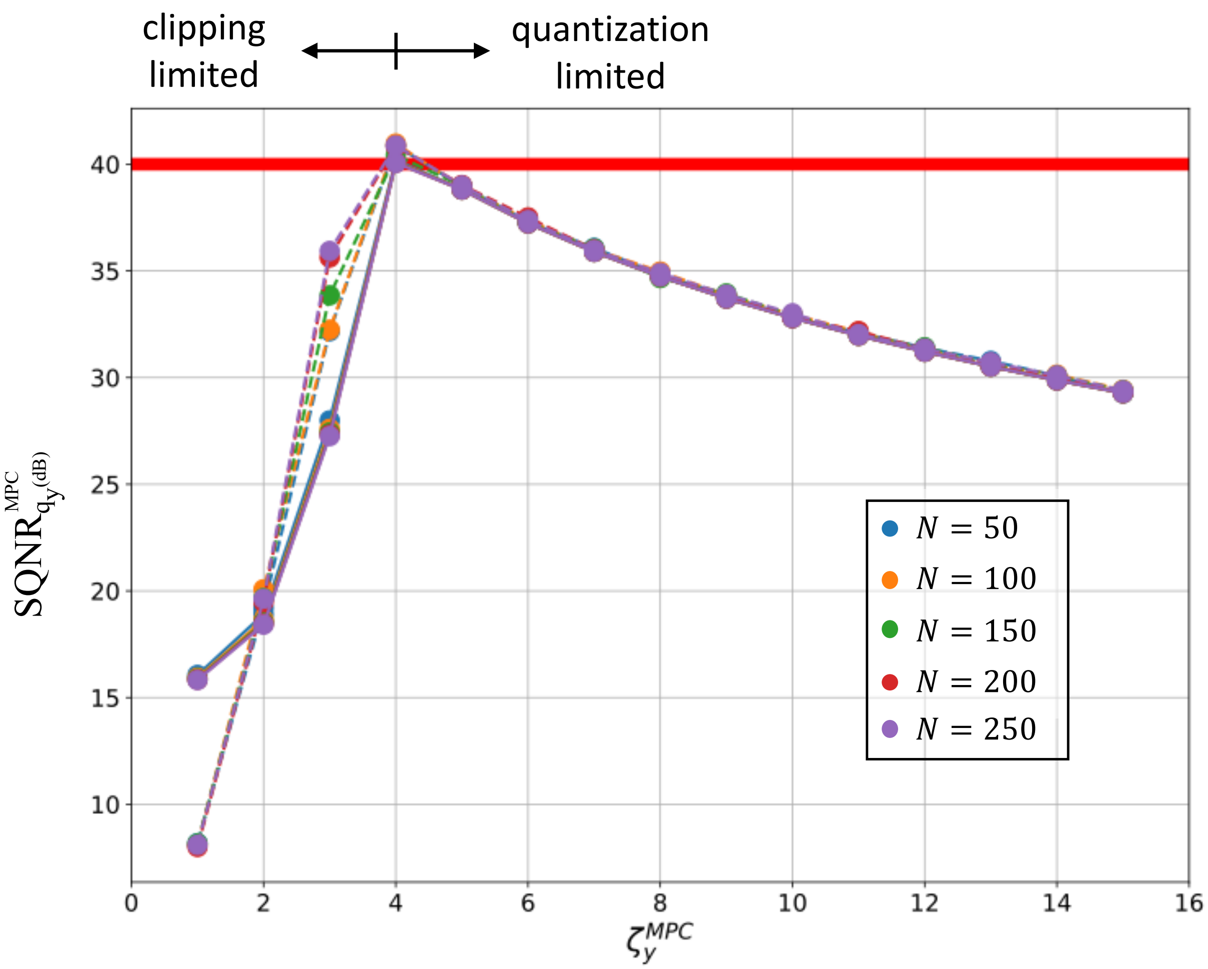
- input: $B_x = 7; \zeta_x = -1.3$ dB $\rightarrow 10^{-\frac{1.3}{10}} = 0.74$ (linear scale);
- weight: $B_w = 7; \zeta_w = 4.8$ dB $\rightarrow\ 10^{\frac{4.8}{10}} = 3.02$ (linear scale);

$$SQNR_{q_{iy}} = 6(B_x + B_w) + 4.8 - [\zeta_x(dB) + \zeta_w(dB)] - 10\log_{10}\left(\frac{2^{2B_x}}{\zeta_x} + \frac{2^{2B_w}}{\zeta_w}\right)$$

$$= 6\times 14 + 4.8 - [-1.3 + 4.8] - 10\log_{10}\left(\frac{2^{14}}{0.74} + \frac{2^{14}}{3.02}\right) = 41 \text{ dB}$$

- assign B_y such that $SQNR_{q_y} \geq 40$ dB so that $SQNR_T \approx 40 - 3 = 37$ dB

Clipping vs. Quantization Noise Trade-off

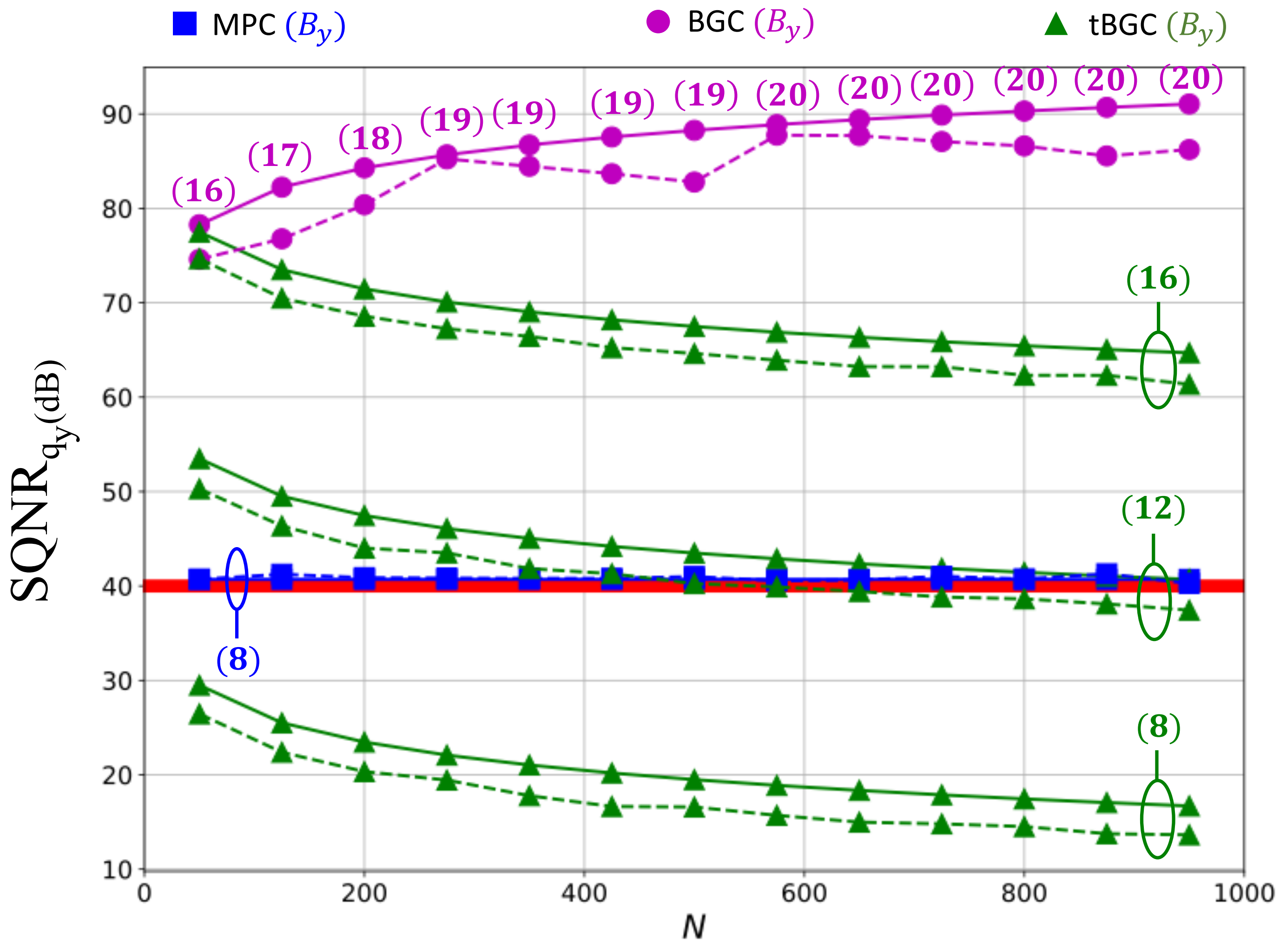


check for $\zeta_y^{MPC} = 4, B_y = 8$: $SQNR_{q_y}^{MPC}(dB) = 6\times 8 + 4.8 - 20\log_{10} 4 - 0.57 = 40.2$ dB

$$SQNR_{q_y}^{MPC}(dB) = 6B_y + 4.8 - \zeta_y^{MPC}(dB) - 10\log_{10}\left(1 + p_c\frac{\sigma_{cc}^2}{\sigma_{q_y}^2}\right)$$

- $B_y^{MPC} = 8$
- $p_c = 2Q(4) = 6.3\times 10^{-5}$ (for ζ_y^{MPC}= 4)
- $\sigma_{cc}^2 = \mathrm{E}\left\{\left||y| - \mathrm{y_c}\right|^2 \Big| |y| > y_c\right\}$
  $= \sigma_{y_o}^2 \times 0.19$ (for $y_c = 4\sigma_{y_o}$)
  → computed using numerical integration
- $\sigma_{q_y}^2 = \frac{y_c^2 2^{-2B_y}}{3} = \frac{\sigma_{y_o}^2 (\zeta_y^{MPC})^2 2^{-2B_y}}{3}$
  $= \sigma_{y_o}^2 \times 8.1\times 10^{-5}$ (for ζ_y^{MPC}= 4 and $B_y^{MPC} = 8$)
- $\rightarrow p_c\frac{\sigma_{cc}^2}{\sigma_{q_y}^2} = 0.14$ (for ζ_y^{MPC}= 4 and $B_y^{MPC} = 8$)
- $\rightarrow\ 10\log_{10}\left(1 + p_c\frac{\sigma_{cc}^2}{\sigma_{q_y}^2}\right) = 0.57$

Comparing MPC and BGC



- MPC achieves the desired $SQNR_{q_y}^* = 40$ dB with minimum precision ($B_y = 8$)
- BGC is a huge overkill → leads to very large accumulator bit widths ($B_y = 16$ to 20)
- tBGC (truncated BGC) needs $B_y = 12$ (still significant)
- Use MPC to assign minimum accumulator/output precision

Summary

- precision reduction is an effective method to reduce DNN complexity
- need to reduce input, weight and output precision of dot products
- quantization effects modeled as additive noise – a practical approximation
- low-precision options – fixed-point and minifloats (low-precision float)
- number representations – sign-magnitude, 2’s complement, log….
- fixed-point dot products – two approaches to handle weight quantization (perturbation model and noise model)
- total SQNR is limited by input quantization noise
- output precision can be assigned using: 1) Bit-Growth Criterion (BGC) (overly conservative); 2) truncated BGC (better); 3) Minimum Precision Criterion (MPC) (best) that achieves the same SQNR as BGC but with much lower precision

Course Web Page

https://courses.grainger.illinois.edu/ece598nsg/fa2020/
https://courses.grainger.illinois.edu/ece498nsu/fa2020/

http://shanbhag.ece.uiuc.edu