

Rotary Position Embeddings (RoPE)

Comprehensive Demo and Analysis

RoPE encodes relative position directly into attention dot products through rotation. Each dimension pair defines a 2D subspace with position-dependent rotation.

Key property: $\langle \text{RoPE}(q,m), \text{RoPE}(k,n) \rangle$ depends only on $(m-n)$, making RoPE a true relative position encoding.

Used by: Llama 1/2/3, Mistral, Qwen, Gemma, DeepSeek, and virtually all modern open-weight LLMs.

Random seed: 42
Number of visualizations: 6
Examples: 6

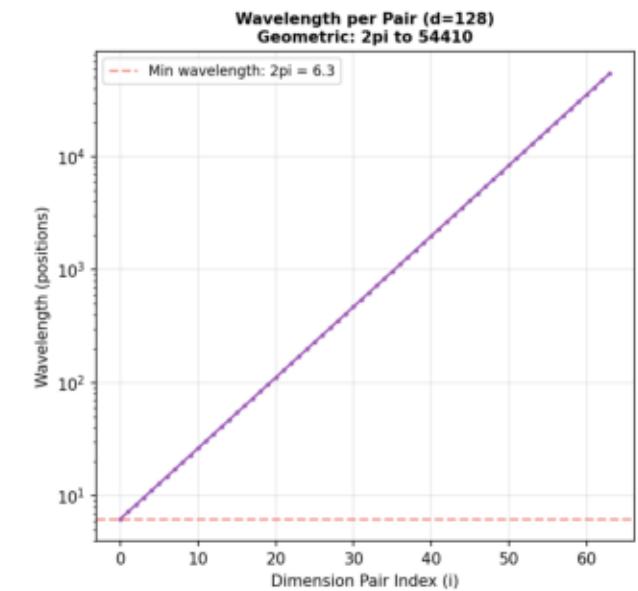
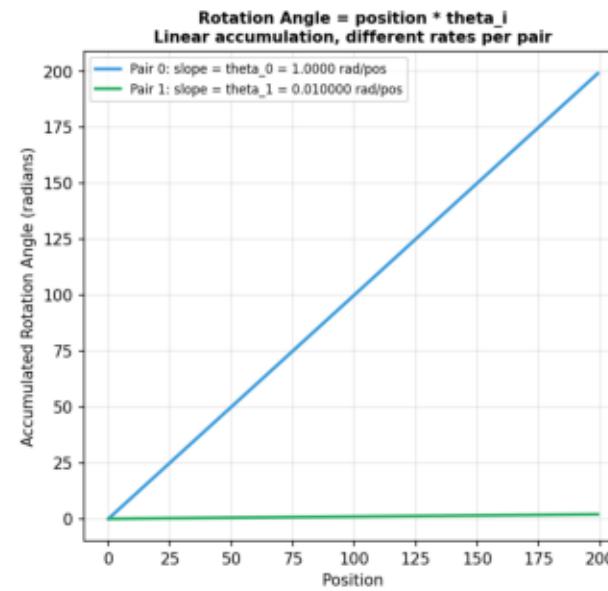
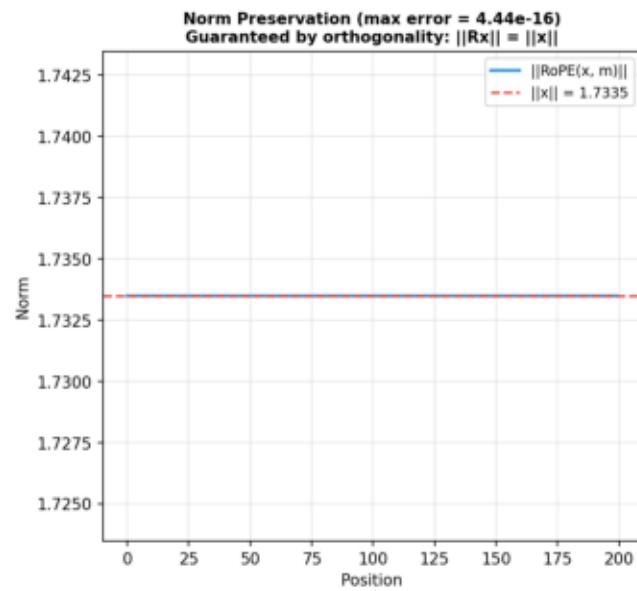
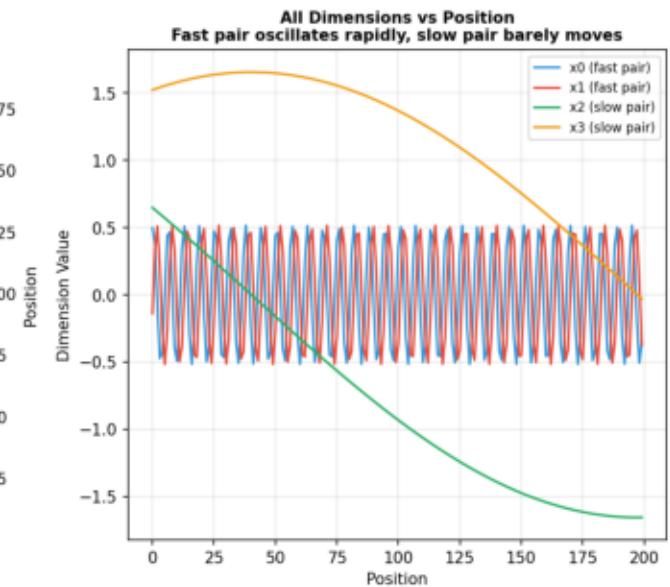
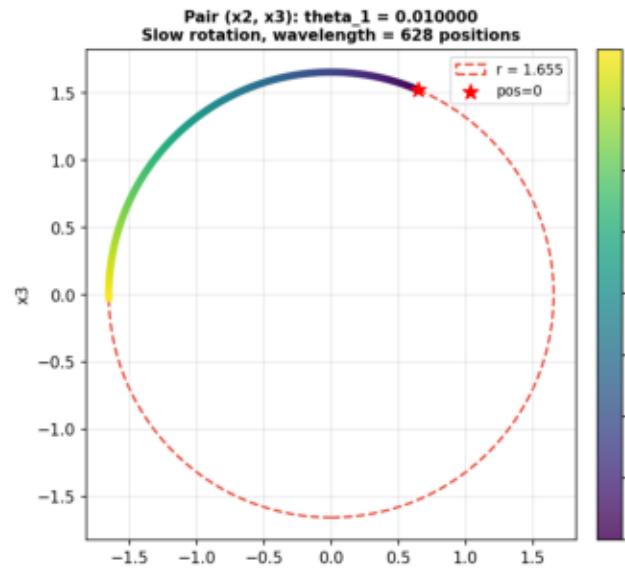
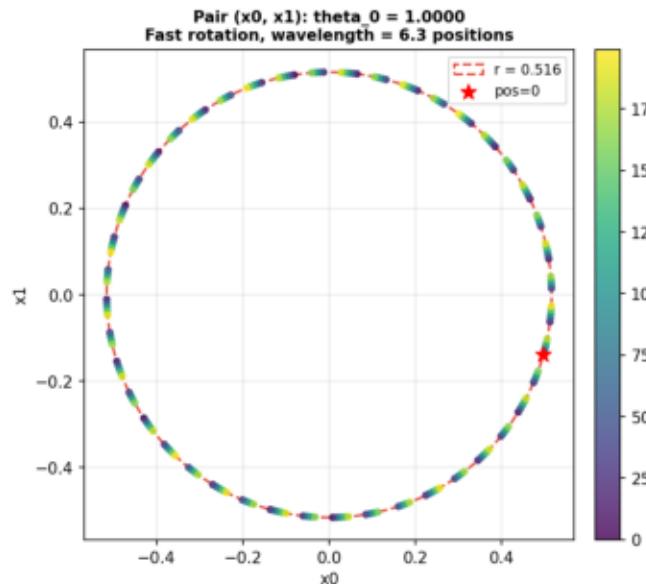
Generated by demo.py

Summary of Findings

1. Rotation Visualization: Each dimension pair $(2i, 2i+1)$ traces a circle in its 2D subspace. Pair 0 rotates fast ($\theta_0 = 1.0$), higher pairs rotate progressively slower. Norm is preserved because rotations are orthogonal transformations.
2. Relative Position Property (CENTERPIECE): The dot product $\langle \text{RoPE}(q,m), \text{RoPE}(k,n) \rangle = q^T R(n-m)$ k depends ONLY on $(m-n)$.
Proof: $R(m)^T R(n) = R(-m)R(n) = R(n-m)$ by angle addition.
Verified empirically: max variation < 1e-12 across all positions.
3. Norm & Orthogonality: $\|\text{RoPE}(x,m)\| = \|x\|$ guaranteed by $R^T R = I$.
 $\det(R) = 1$ (proper rotation). $R(m)R(n) = R(m+n)$ (composition).
 $R(m)R(-m) = I$ (inverse). All analytically exact from $\cos^2 + \sin^2 = 1$.
4. RoPE vs Sinusoidal PE: Both use $\theta_i = 10000^{(-2i/d)}$.
Sinusoidal is ADDITIVE (position added to content), creating cross-terms that depend on absolute position. RoPE is MULTIPLICATIVE ($q' = R(m)q$), giving a pure relative position encoding. This is the key advantage.
5. Attention Impact: With identical token embeddings, RoPE creates position-dependent attention patterns (weights vary by relative position). Without RoPE, attention is uniform. Patterns are shift-invariant (shifting all positions preserves relative distances).
6. Context Extension: Larger θ_{base} stretches wavelengths, extending effective context. Llama 3 uses $\theta=500K$ vs standard 10K. NTK-aware scaling preserves high-freq (local) while stretching low-freq (long-range) -- can extend context without retraining.

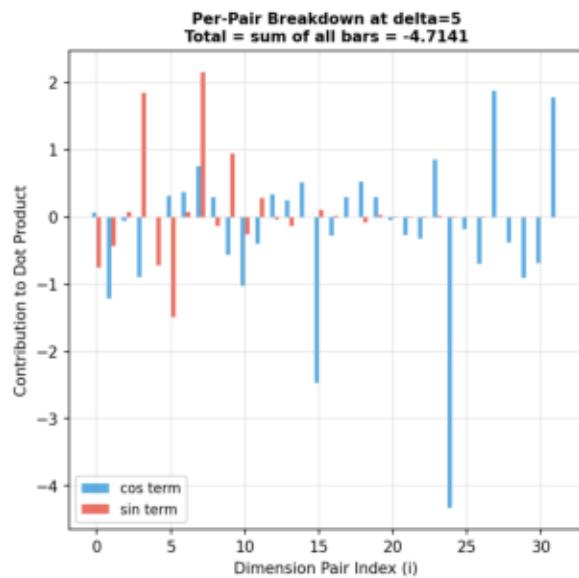
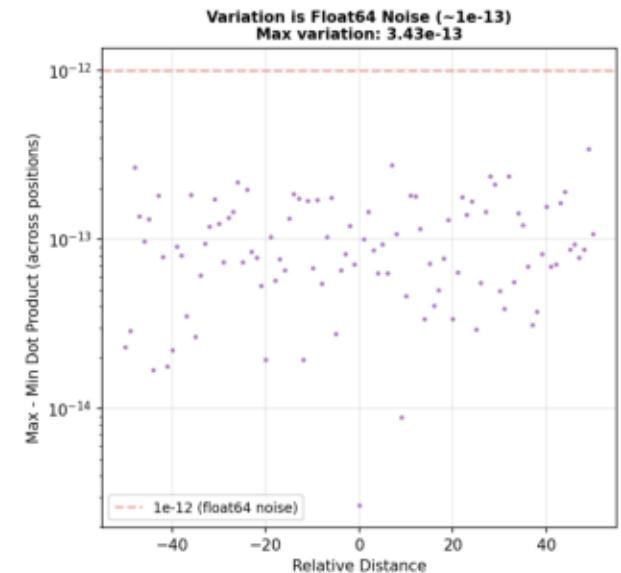
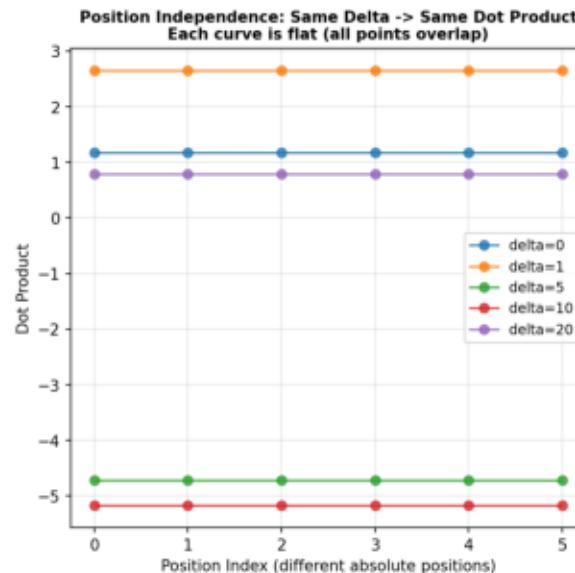
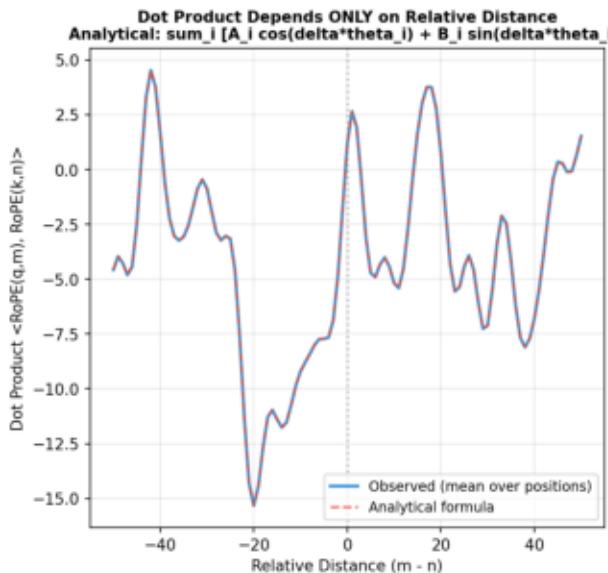
Example 1: Rotation Visualization -- Circles in 2D Subspaces

RoPE Rotation Visualization: Each Dimension Pair Traces a Circle



Example 2: Relative Position Property -- The Key Theorem

RoPE Relative Position Property: Dot Product Depends Only on (m-n)



ANALYTICAL DERIVATION

Given: $q' = R(m)q, k' = R(n)k$

$$\langle q', k' \rangle = \langle R(m)q \rangle^T \langle R(n)k \rangle$$

$$= q^T R(m)^T R(n) k$$

$$= q^T R(-m) R(n) k \quad [R^T = R^{-1} = R(-m)]$$

$$= q^T R(n-m) k \quad [R(a)R(b) = R(a+b)]$$

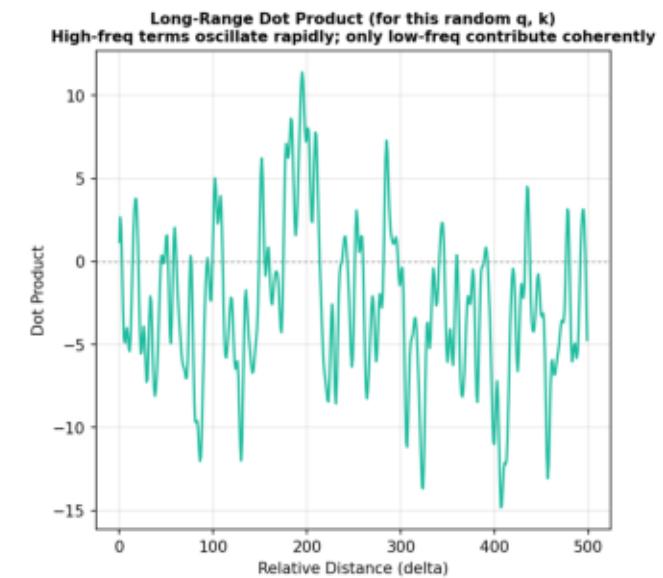
Per pair i:

$$(q_{\{2i\}}k_{\{2i\}} + q_{\{2i+1\}}k_{\{2i+1\}}) \cos((m-n)\theta_i)$$

$$+ (q_{\{2i\}}k_{\{2i+1\}} - q_{\{2i+1\}}k_{\{2i\}}) \sin((m-n)\theta_i)$$

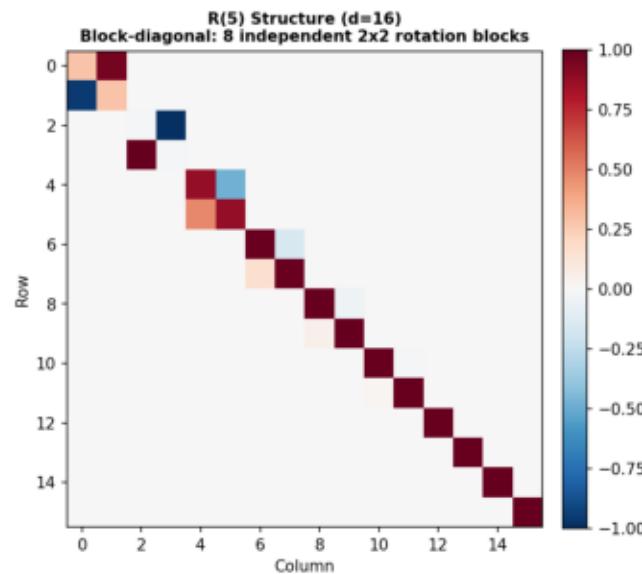
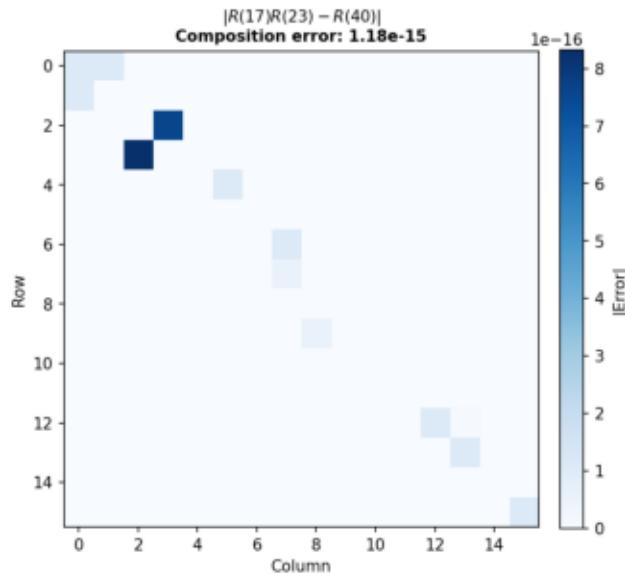
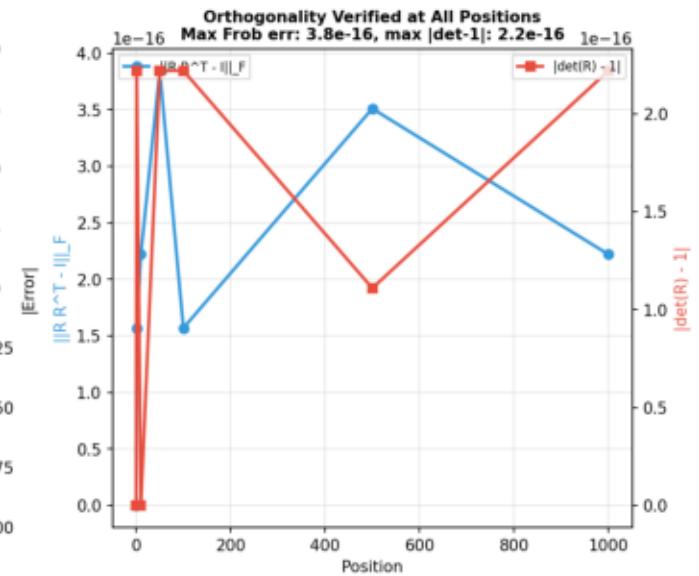
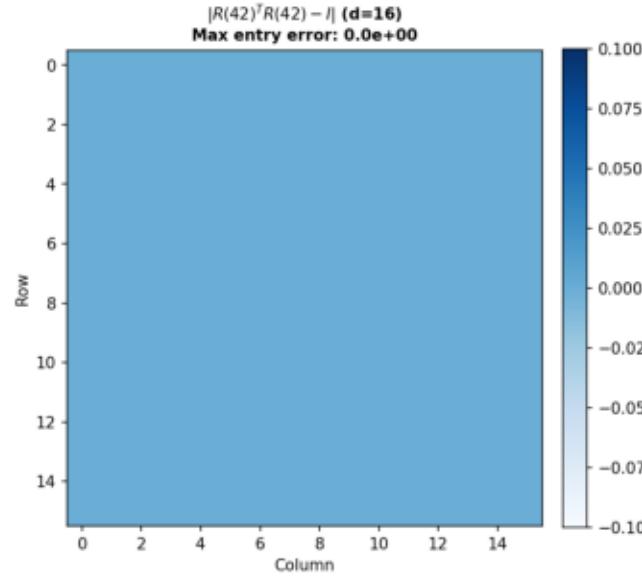
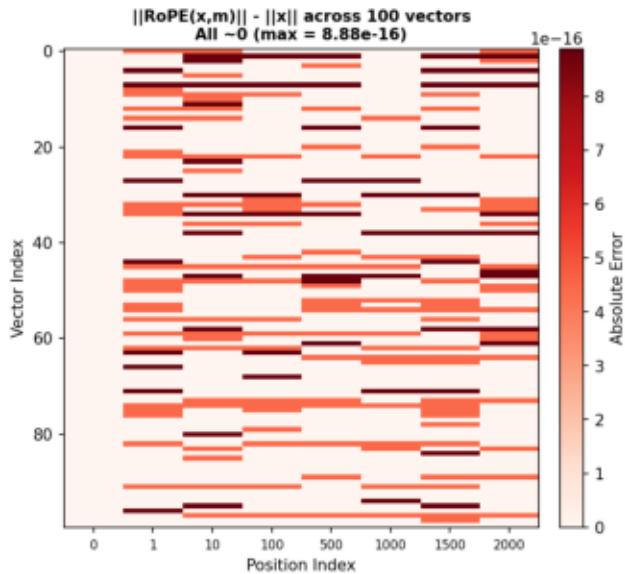
KEY: depends on $(m-n)$, q , k only.
NOT on m or n individually.

This is why RoPE is a RELATIVE position encoding despite using ABSOLUTE positions in the rotation.



Example 3: Norm Preservation & Orthogonality

RoPE Orthogonality: $R^T R = I$, $\det(R) = 1$, $R(m)R(n) = R(m+n)$, $\|Rx\| = \|x\|$



ROTATION MATRIX PROPERTIES
=====

R is 16x16 block-diagonal with 8 blocks

Each 2×2 block $R_i(m)$:
 $[\cos(m^*\theta_i), -\sin(m^*\theta_i)],$
 $[\sin(m^*\theta_i), \cos(m^*\theta_i)]$

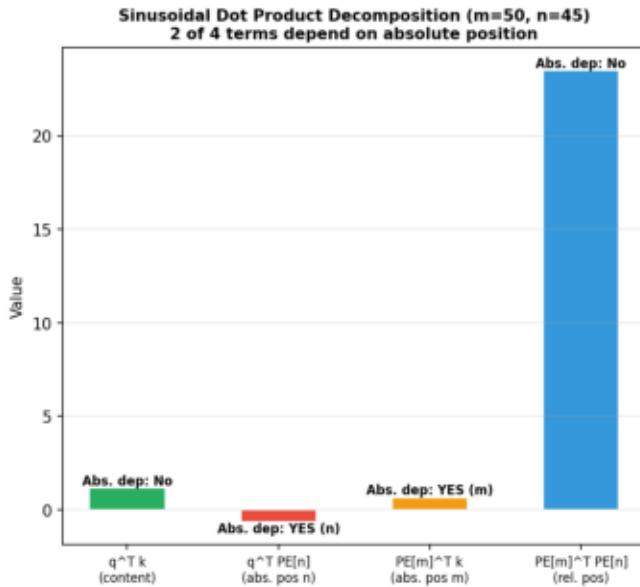
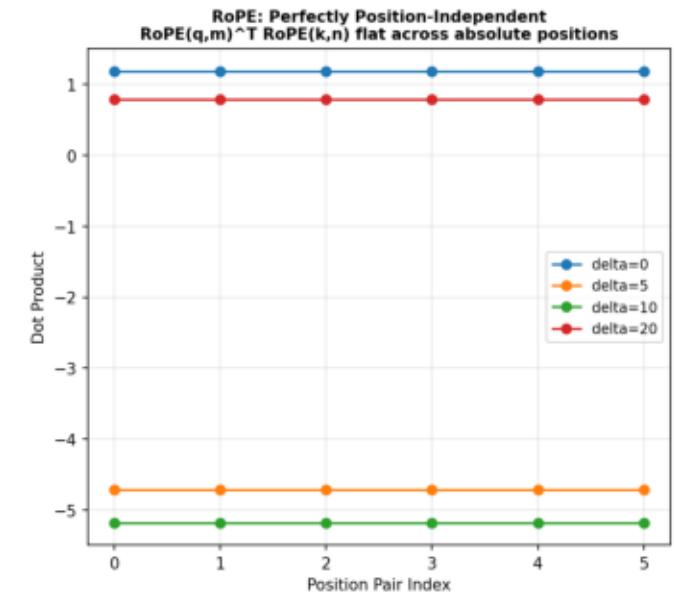
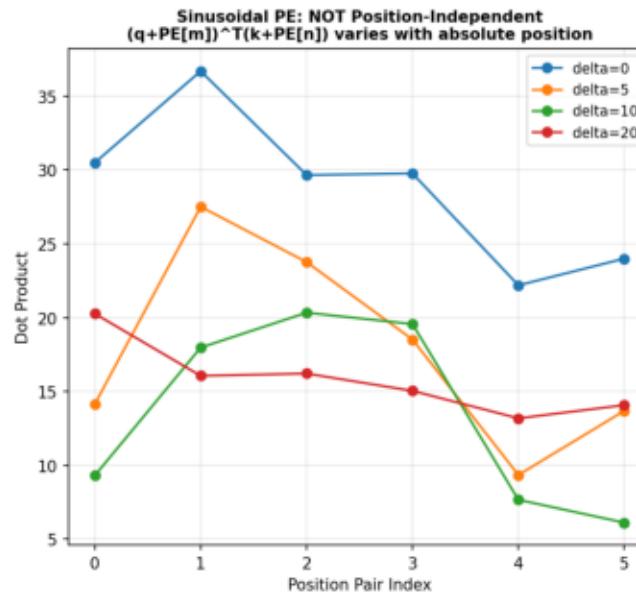
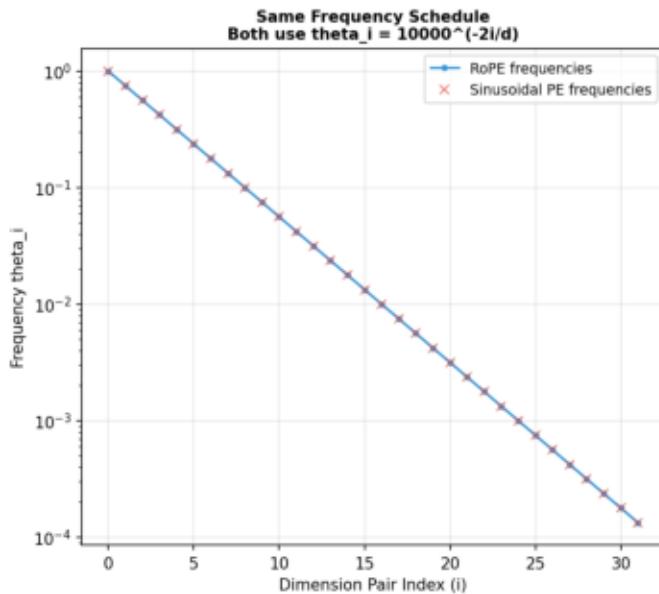
Guaranteed properties:

1. Orthogonality: $R^T R = I$
Verified: $\|R^T R - I\| < 4e-16$
2. Proper rotation: $\det(R) = 1$
Verified: $|\det - 1| < 2e-16$
3. Composition: $R(m)R(n) = R(m+n)$
Verified: error = $1e-15$
4. Inverse: $R(m)R(-m) = I$
Verified: error = $4e-16$
5. Norm preservation: $\|Rx\| = \|x\|$
Verified: max error = $9e-16$

All from $\cos^2 + \sin^2 = 1$.

Example 4: RoPE vs Sinusoidal PE -- Additive vs Multiplicative

RoPE vs Sinusoidal PE: Same Frequencies, Fundamentally Different Application



COMPARISON: ADDITIVE vs MULTIPLICATIVE

SINUSOIDAL PE (Additive):

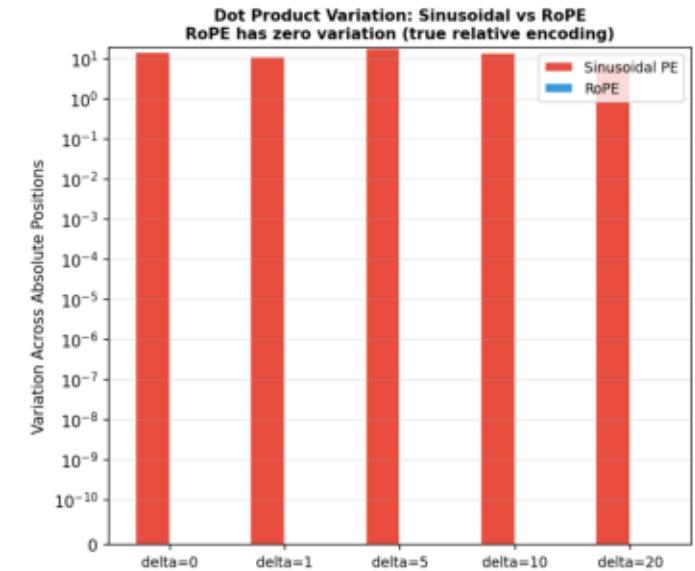
```

input' = input + PE[position]
q = input' @ W_Q
k = input' @ W_K
score = q^T k
      = (x+PE[m])^T W_Q^T W_K (x+PE[n])
--> cross-terms depend on abs. position
  
```

RoPE (Multiplicative):

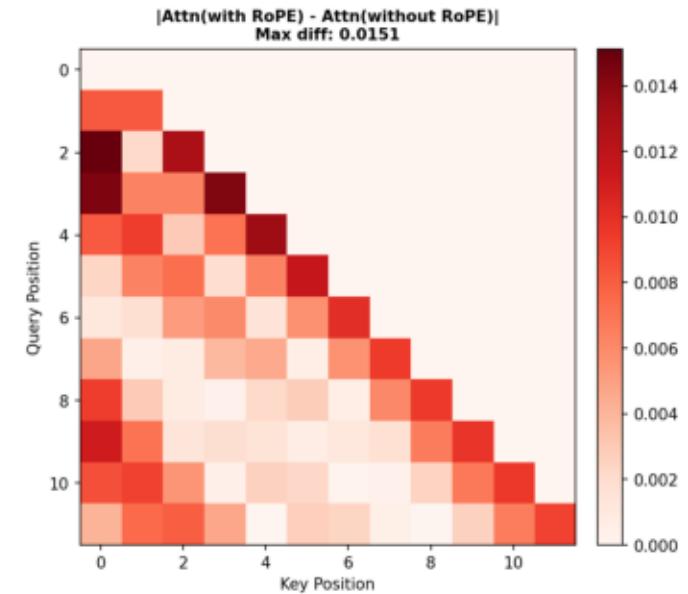
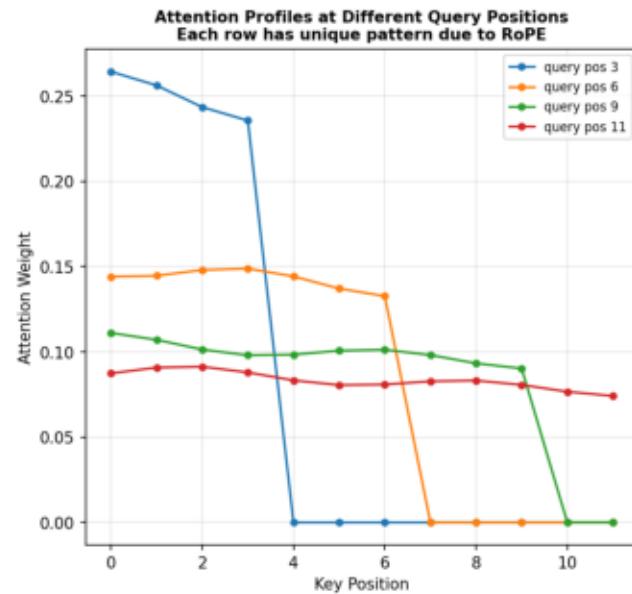
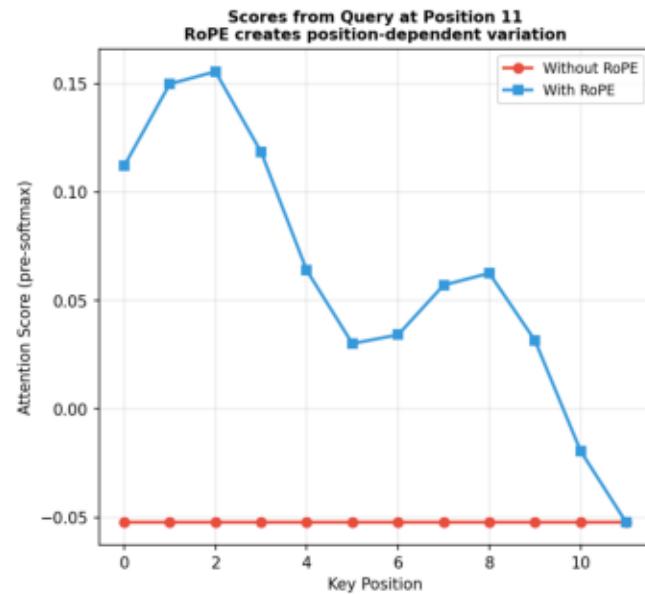
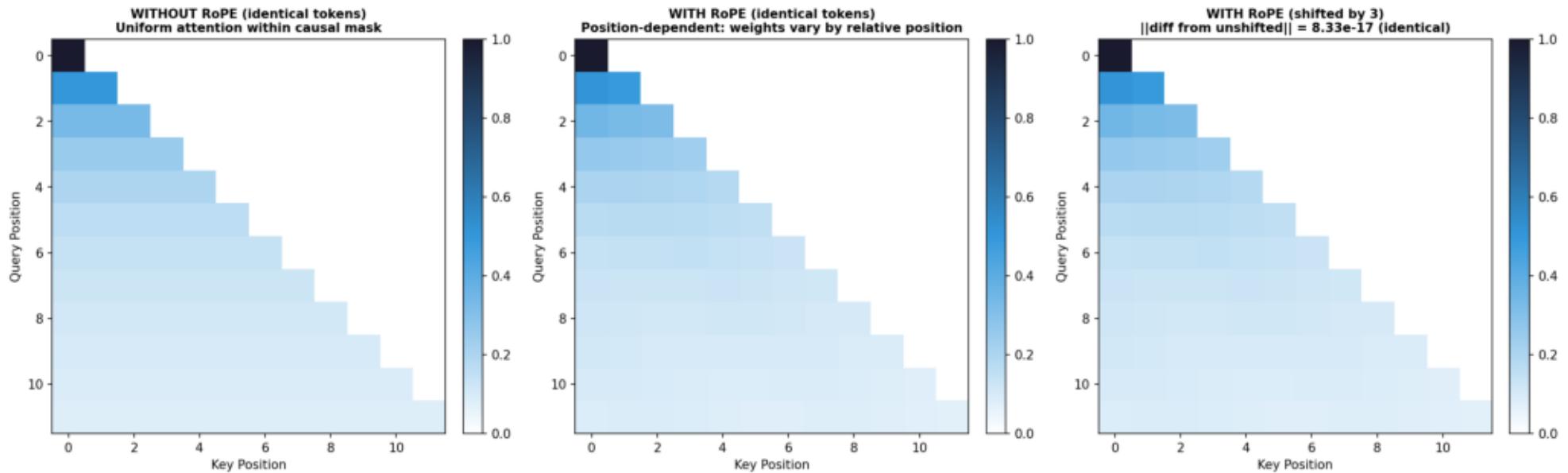
```

q = input @ W_Q
k = input @ W_K
q' = R(m) @ q      (rotate AFTER proj.)
k' = R(n) @ k
score = q'^T k'
      = q^T R(m)^T R(n) k
      = q^T R(n-m) k
--> depends on (n-m) only!
  
```



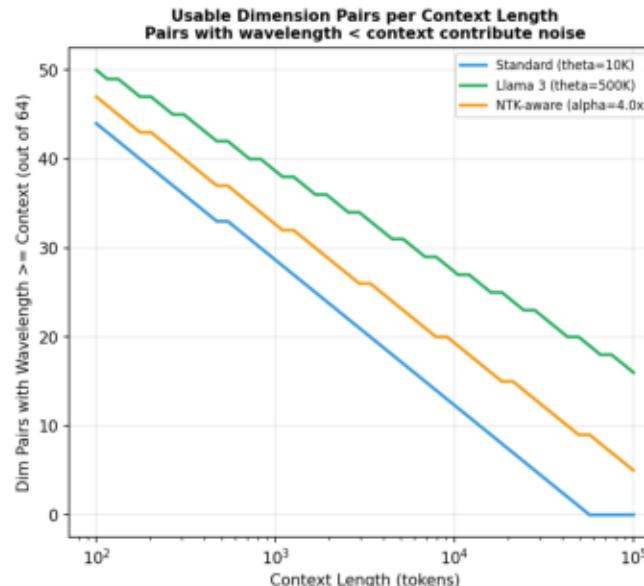
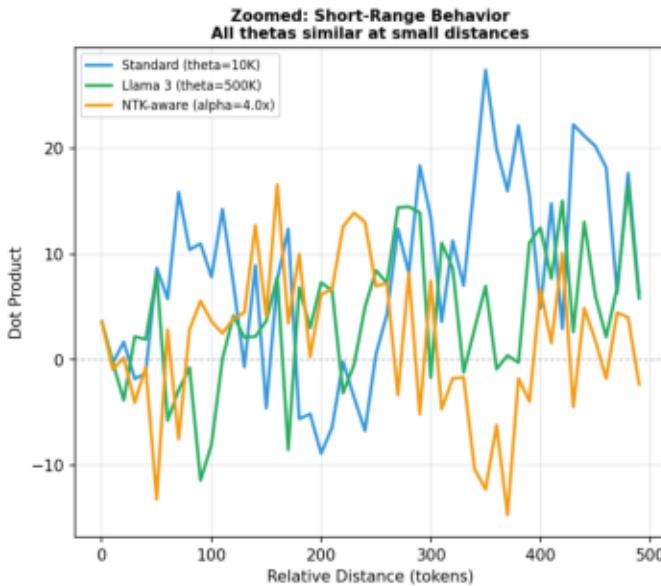
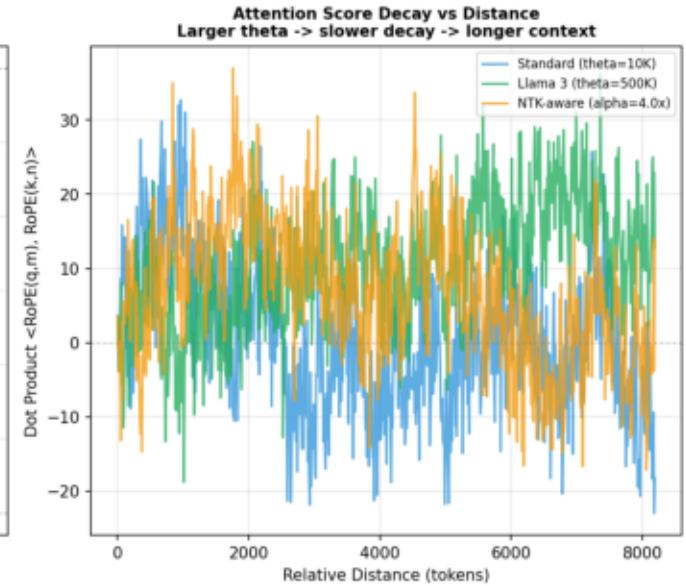
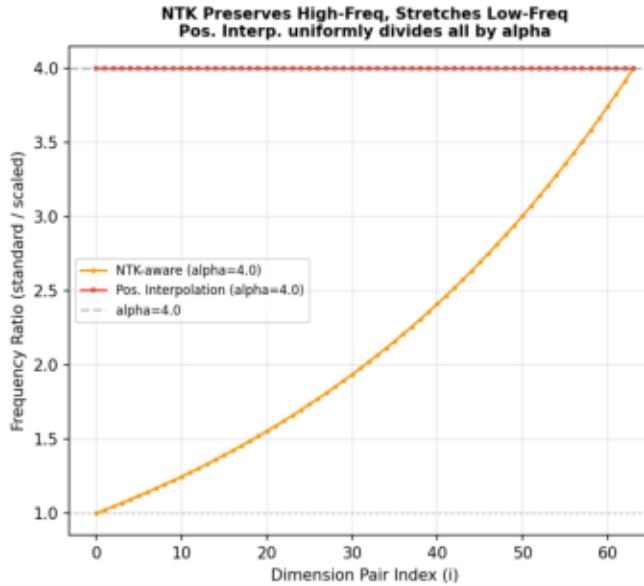
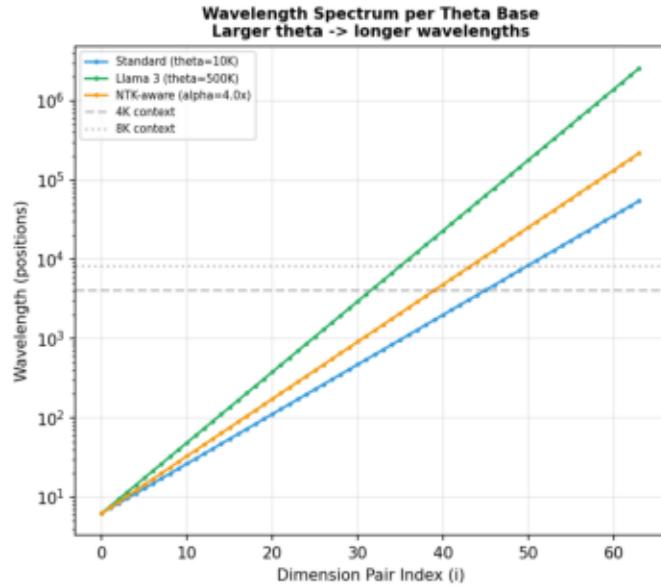
Example 5: Impact on Attention Patterns

Impact of RoPE on Attention: Position-Dependent Patterns from Identical Inputs



Example 6: Context Extension via Theta Scaling

Context Extension via Theta Scaling: Standard vs Llama 3 vs NTK-Aware



CONTEXT EXTENSION METHODS

Standard (theta=10000):
Effective context: ~4K-8K tokens
Used by: Llama 1/2, RoFormer

Higher base (theta=500000):
Effective context: ~128K tokens
Used by: Llama 3, Qwen 2
Simple but requires retraining

NTK-aware (alpha=4.0, theta'=40890):
 $\theta' = \theta \cdot \alpha^{(d/(d-2))}$
Preserves high-freq (local) resolution
Stretches low-freq (long-range) only
Can be applied WITHOUT retraining

Position interpolation (alpha=4.0):
Divides ALL positions by alpha
Simpler but degrades local resolution
Requires fine-tuning for good results