An Algebraic Framework for Updatable and Universal (zk)SNARKs

Carla Ràfols and *Arantxa Zapico*

4th ZKProof Workshop

Universitat Pompeu Fabra Barcelona

Obra Social Fundación "la Caixa"

Pairing-Based (zk)SNARKs

State of the art

Interactive Proof-Systems [GMR89]

Pairing-Based (zk)SNARKs

State of the art

Interactive Proof-Systems [GMR89] $\longrightarrow$ ZK proofs for all NP [GMW] $\longrightarrow$ $\ldots$ $\longrightarrow$ Succinct arguments without PCPs [Gro10] $\longrightarrow$ QAPs [GGPR13] & Pinnocchio [PGHR13] $\longrightarrow$ ZeroCash

Pairing-Based (zk)SNARKs

State of the art

Interactive Proof-Systems [GMR89]$\longrightarrow$ ZK proofs for all NP [GMW] $\longrightarrow$ ... $\longrightarrow$ Succinct arguments without PCPs [Gro10] $\longrightarrow$ QAPs [GGPR13] & Pinnocchio [PGHR13] $\longrightarrow$ ZeroCash $\longrightarrow$ Most efficient zk-SNARK [Gro16]

Pairing-Based (zk)SNARKs

State of the art

Interactive Proof-Systems [GMR89]$\longrightarrow$ ZK proofs for all NP [GMW] $\longrightarrow$ . . . $\longrightarrow$ Succinct arguments without PCPs [Gro10] $\longrightarrow$ QAPs [GGPR13] & Pinnocchio [PGHR13] $\longrightarrow$ ZeroCash $\longrightarrow$ Most efficient zk-SNARK [Gro16]

Trusted Setup!!!

Pairing-Based (zk)SNARKs

State of the art

Interactive Proof-Systems [GMR89] ⟶ ZK proofs for all NP [GMW] ⟶ . . . ⟶ Succinct arguments without PCPs [Gro10] ⟶ QAPs [GGPR13] & Pinnocchio [PGHR13] ⟶ ZeroCash ⟶ Most efficient zk-SNARK [Gro16]

Trusted Setup!!!

Multiparty Computation (Zcash Ceremony)

Pairing-Based (zk)SNARKs

State of the art

Interactive Proof-Systems [GMR89] $\longrightarrow$ ZK proofs for all NP [GMW] $\longrightarrow$ . . . $\longrightarrow$ Succinct arguments without PCPs [Gro10] $\longrightarrow$ QAPs [GGPR13] & Pinnocchio [PGHR13] $\longrightarrow$ ZeroCash $\longrightarrow$ Most efficient zk-SNARK [Gro16]

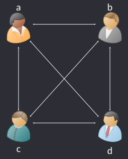
Trusted Setup!!!

Multiparty Computation (Zcash Ceremony)

One ceremony per circuit !!!

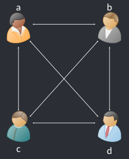
Updatable and Universal SNARKs

- Multiparty Computation Model:

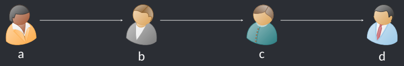


Updatable and Universal SNARKs

- Multiparty Computation Model:

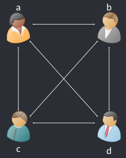


- Updatable Model:

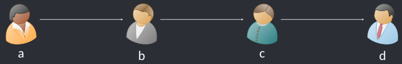


Updatable and Universal SNARKs

- Multiparty Computation Model:



- Updatable Model:



Circuit Specific vs. Universal

Motivation

Ishai's wisdom

Motivation

Ishai's wisdom

Great we have a lot of research in zero-knowledge, but comparison is still difficult

Motivation

Ishai's wisdom

Great we have a lot of research in zero-knowledge, but comparison is still difficult

- Multitude of application scenarios, implementation details, efficiency desiderata, cryptographic assumptions, and trust models,

Motivation

Ishai's wisdom

Great we have a lot of research in zero-knowledge, but comparison is still difficult

- Multitude of application scenarios, implementation details, efficiency desiderata, cryptographic assumptions, and trust models,
- It is all packed: makes it difficult to apply a mix-and-match approach for finding the best combination of the underlying ideas in the context of a given application.

Motivation

Ishai's wisdom

Great we have a lot of research in zero-knowledge, but comparison is still difficult

- Multitude of application scenarios, implementation details, efficiency desiderata, cryptographic assumptions, and trust models,
- It is all packed: makes it difficult to apply a mix-and-match approach for finding the best combination of the underlying ideas in the context of a given application.

"This calls for a modular approach that allows for an easier navigation in the huge design space. A higher level of modularity and abstraction is useful (...) "

Updatable and Universal (zk)SNARKs

Common Design Principle

Updatable and Universal (zk)SNARKs

Common Design Principle

AHP/
PHP/
...

Updatable and Universal (zk)SNARKs

Common Design Principle

$\boxed{\text{AHP/ PHP/ } \ldots} + \boxed{\text{Polynomial Commitment}}$

Updatable and Universal (zk)SNARKs

Common Design Principle

$$\boxed{\begin{array}{c}\text{AHP/}\\ \text{PHP/}\\ \ldots\end{array}} + \boxed{\begin{array}{c}\text{Polynomial}\\ \text{Commitment}\end{array}} = \boxed{\text{zkSNARK}}$$

Updatable and Universal (zk)SNARKs

Common Design Principle

$\boxed{\text{AHP/ PHP/ } \ldots} + \boxed{\text{Polynomial Commitment}} = \boxed{\text{zkSNARK}}$

Holographic:

Updatable and Universal (zk)SNARKs

Common Design Principle

AHP/ PHP/ …	+	Polynomial Commitment	=	zkSNARK

Holographic:

- *Indexer* computes relation-dependent polynomials

Updatable and Universal (zk)SNARKs

Common Design Principle

$$\boxed{\begin{array}{c}\text{AHP/}\\ \text{PHP/}\\ \ldots\end{array}} + \boxed{\begin{array}{c}\text{Polynomial}\\ \text{Commitment}\end{array}} = \boxed{\text{zkSNARK}}$$

Holographic:

- *Indexer* computes relation-dependent polynomials
- *Prover's* messages include polynomials

Updatable and Universal (zk)SNARKs

Common Design Principle

$\boxed{\text{AHP/ PHP/ } \ldots} + \boxed{\text{Polynomial Commitment}} = \boxed{\text{zkSNARK}}$

Holographic:

- *Indexer* computes relation-dependent polynomials
- *Prover's* messages include polynomials
- *Verifier* has oracle access to both sets of polynomials, can do degree checks, etc.

Updatable and Universal (zk)SNARKs

Common Design Principle

$$\boxed{\begin{array}{c}\text{AHP/}\\ \text{PHP/}\\ \dots\end{array}} + \boxed{\begin{array}{c}\text{Polynomial}\\ \text{Commitment}\end{array}} = \boxed{\text{zkSNARK}}$$

Holographic:

- *Indexer* computes relation-dependent polynomials
- *Prover's* messages include polynomials
- *Verifier* has oracle access to both sets of polynomials, can do degree checks, etc.

Sonic[MBKM19] — Plonk[GWC19] — Marlin[CHMMVW20] — Lunar[CFFQH20] — Claymore[SZ21]

Motivation

This proposal

Motivation

This proposal

¿Can we break down further the information theoretic component?

Organization

- Break Circuit Satisfiability in three main algebraic components:

Organization

- Break Circuit Satisfiability in three main algebraic components:
  - Hadamard Product

Organization

- Break Circuit Satisfiability in three main algebraic components:
  - Hadamard Product
  - Inner Product

Organization

- Break Circuit Satisfiability in three main algebraic components:
  - Hadamard Product
  - Inner Product
  - Verifiable Subspace Sampling

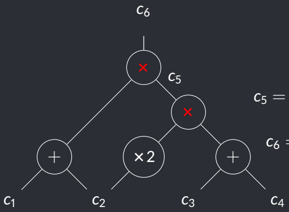
Organization

- Break Circuit Satisfiability in three main algebraic components:
  - Hadamard Product
  - Inner Product
  - Verifiable Subspace Sampling

- Definition of Verifiable Subspace Sampling

Groth16: Overview Example

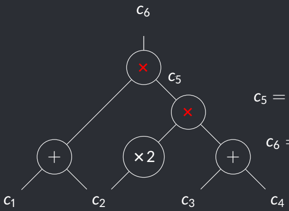
- Constraint system:



$$\begin{aligned} c_5 &= (2c_2)(c_3+c_4) \\ c_6 &= (c_1+c_2)c_5 \end{aligned} \iff (2c_2, c_1+c_2) \circ (c_3+c_4, c_5) = (c_5, c_6)$$

Groth16: Overview Example

- Constraint system:

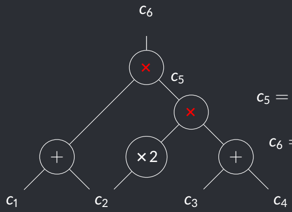


$$\begin{aligned} c_5 &= (2c_2)(c_3 + c_4) \\ c_6 &= (c_1 + c_2)c_5 \end{aligned} \iff (2c_2, c_1 + c_2) \circ (c_3 + c_4, c_5) = (c_5, c_6)$$

- Argument: has two main building blocks

Groth16: Overview Example

- Constraint system:

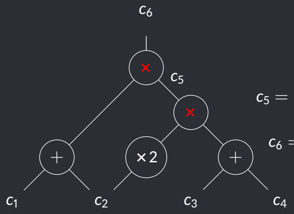


$$c_5 = (2c_2)(c_3 + c_4) \qquad c_6 = (c_1 + c_2)c_5 \iff (2c_2, c_1 + c_2) \circ (c_3 + c_4, c_5) = (c_5, c_6)$$

- Argument: has two main building blocks
  1. Hadamard product

Groth16: Overview Example

- Constraint system:

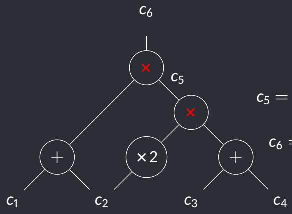


$$c_5 = (2c_2)(c_3 + c_4)$$
$$c_6 = (c_1 + c_2)c_5$$
$$\iff (2c_2, c_1 + c_2) \circ (c_3 + c_4, c_5) = (c_5, c_6)$$

- Argument: has two main building blocks
  1. Hadamard product Universal SRS

Groth16: Overview Example

- Constraint system:

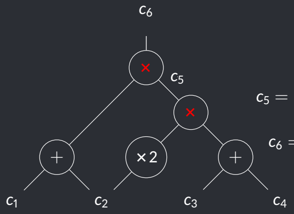


$$c_5 = (2c_2)(c_3 + c_4) \qquad c_6 = (c_1 + c_2)c_5 \iff (2c_2, c_1 + c_2) \circ (c_3 + c_4, c_5) = (c_5, c_6)$$

- Argument: has two main building blocks
  1. Hadamard product Universal SRS
  2. Linear Relations

Groth16: Overview Example

- Constraint system:



$$\begin{array}{c} c_5 = (2c_2)(c_3 + c_4) \\ c_6 = (c_1 + c_2)c_5 \end{array} \iff (2c_2, c_1 + c_2) \circ (c_3 + c_4, c_5) = (c_5, c_6)$$

- Argument: has two main building blocks
  1. Hadamard product Universal SRS
  2. Linear Relations Circuit-Dependent SRS

Main Tool: “Compressed” Linear Algebra

Hadamard Product

[1] Completeness, soundness: $\mathcal{R}$ arbitrary set, efficiency: $\mathcal{R}$ multiplicative subgroup.

Main Tool: “Compressed” Linear Algebra

Hadamard Product

Let $\mathscr{R} = \{r_1, \ldots, r_m\} \subset \mathbb{F}_p^*$.

[1] Completeness, soundness: $\mathscr{R}$ arbitrary set, efficiency: $\mathscr{R}$ multiplicative subgroup.

Main Tool: "Compressed" Linear Algebra

Hadamard Product

Let $\mathscr{R} = \{r_1, \ldots, r_m\} \subset \mathbb{F}_p^*$.

$$\lambda_i(X) = \prod_{j \neq i} \frac{(X - r_j)}{(r_i - r_j)},$$

[1] Completeness, soundness: $\mathscr{R}$ arbitrary set, efficiency: $\mathscr{R}$ multiplicative subgroup.

Main Tool: “Compressed” Linear Algebra

Hadamard Product

Let $\mathscr{R} = \{r_1, \ldots, r_m\} \subset \mathbb{F}_p^*$.

$$\lambda_i(X) = \prod_{j \neq i} \frac{(X - r_j)}{(r_i - r_j)}, \qquad t(X) = \prod_j (X - r_j)$$

[1] Completeness, soundness: $\mathscr{R}$ arbitrary set, efficiency: $\mathscr{R}$ multiplicative subgroup.

Main Tool: “Compressed” Linear Algebra

Hadamard Product

Let $\mathscr{R} = \{r_1, \ldots, r_m\} \subset \mathbb{F}_p^*$.

$$\lambda_i(X) = \prod_{j \neq i} \frac{(X - r_j)}{(r_i - r_j)}, \qquad t(X) = \prod_j (X - r_j)$$

Linear Algebra World	Polynomial World
$\vec{y} = (y_1, \ldots, y_m)$	$Y(X) = \sum_{i=1}^{m} y_i \lambda_i(X)$

[1] Completeness, soundness: $\mathscr{R}$ arbitrary set, efficiency: $\mathscr{R}$ multiplicative subgroup.

Main Tool: "Compressed" Linear Algebra

Hadamard Product

Let $\mathscr{R} = \{r_1, \ldots, r_m\} \subset \mathbb{F}_p^*$.

$$\lambda_i(X) = \prod_{j \neq i} \frac{(X - r_j)}{(r_i - r_j)}, \qquad t(X) = \prod_j (X - r_j)$$

Linear Algebra World	Polynomial World
$\vec{y} = (y_1, \ldots, y_m)$	$Y(X) = \sum_{i=1}^{m} y_i \lambda_i(X)$
$\vec{z} = \vec{w} \circ \vec{y}$	$t(X) \mid Z(X) - Y(X)W(X)$

[1]Completeness, soundness: $\mathscr{R}$ arbitrary set, efficiency: $\mathscr{R}$ multiplicative subgroup.

Main Tool: "Compressed" Linear Algebra

Hadamard Product

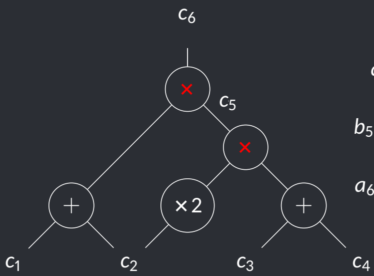
Let $\mathscr{R} = \{r_1, \ldots, r_m\} \subset \mathbb{F}_p^*$.

$$\lambda_i(X) = \prod_{j \neq i} \frac{(X - r_j)}{(r_i - r_j)}, \qquad t(X) = \prod_j (X - r_j)$$

Linear Algebra World	Polynomial World
$\vec{y} = (y_1, \ldots, y_m)$	$Y(X) = \sum_{i=1}^{m} y_i \lambda_i(X)$
$\vec{z} = \vec{w} \circ \vec{y}$	$t(X) \mid Z(X) - Y(X)W(X)$ [1]

[1] Completeness, soundness: $\mathscr{R}$ arbitrary set, efficiency: $\mathscr{R}$ multiplicative subgroup.

Universal SRS: Constraint System



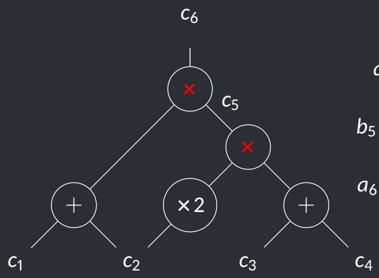
$$a_5 = (2c_2)$$

$$b_5 = (c_3 + c_4) \qquad c_5 = a_5 b_5$$

$$a_6 = (c_1 + c_2) \qquad c_6 = a_6 b_6$$

$$b_6 = c_5$$

Universal SRS: Constraint System



$$a_5 = (2c_2)$$

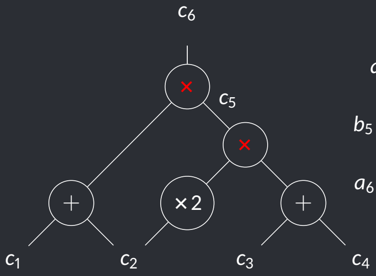
$$b_5 = (c_3 + c_4) \qquad c_5 = a_5 b_5$$

$$a_6 = (c_1 + c_2) \qquad c_6 = a_6 b_6$$

$$b_6 = c_5$$

a_i's left inputs, b_i's right inputs, c_i's outputs.

Universal SRS: Constraint System



$$a_5 = (2c_2)$$

$$b_5 = (c_3 + c_4) \qquad c_5 = a_5 b_5$$

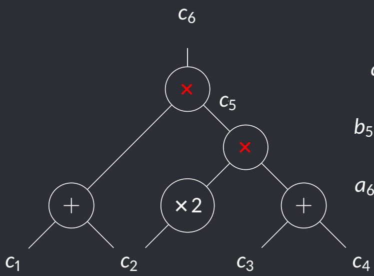
$$a_6 = (c_1 + c_2) \qquad c_6 = a_6 b_6$$

$$b_6 = c_5$$

a_i's left inputs, b_i's right inputs, c_i's outputs.

Hadamard product: $(a_5, a_6) \circ (b_5, b_6) = (c_5, c_6)$

Universal SRS: Constraint System



$$a_5 = (2c_2)$$

$$b_5 = (c_3 + c_4) \qquad c_5 = a_5 b_5$$

$$a_6 = (c_1 + c_2) \qquad c_6 = a_6 b_6$$

$$b_6 = c_5$$

a_i's left inputs, b_i's right inputs, c_i's outputs.

Hadamard product: $(a_5, a_6) \circ (b_5, b_6) = (c_5, c_6)$

Linear relations:

$$\begin{pmatrix} a_5 \\ a_6 \end{pmatrix} = \begin{pmatrix} 0 & 2 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \end{pmatrix} \vec{c}, \quad \begin{pmatrix} b_5 \\ b_6 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \vec{c},$$

Universal SRS: Constraint System

Universal SRS: Constraint System

- Hadamard Product Relation: $\vec{a} \circ \vec{b} = \vec{c}$

Universal SRS: Constraint System

- Hadamard Product Relation: $\vec{a} \circ \vec{b} = \vec{c}$

- Linear Relations: $\vec{a} = \mathbf{F}\vec{c} \quad \text{and} \quad \vec{b} = \mathbf{G}\vec{c},$

Universal SRS: Constraint System

- Hadamard Product Relation: $\vec{a} \circ \vec{b} = \vec{c}$
- Linear Relations: $\vec{a} = \mathbf{F}\vec{c} \quad \text{and} \quad \vec{b} = \mathbf{G}\vec{c}$, or equivalently

$$\begin{pmatrix} \mathbf{I} & \mathbf{0} & -\mathbf{F} \end{pmatrix} \begin{pmatrix} \vec{a} \\ \vec{b} \\ \vec{c} \end{pmatrix} = \mathbf{0}, \qquad \begin{pmatrix} \mathbf{0} & \mathbf{I} & -\mathbf{G} \end{pmatrix} \begin{pmatrix} \vec{a} \\ \vec{b} \\ \vec{c} \end{pmatrix} = \mathbf{0}$$

Universal SRS: Constraint System

- Hadamard Product Relation: $\vec{a} \circ \vec{b} = \vec{c}$
- Linear Relations: $\vec{a} = \mathbf{F}\vec{c} \quad \text{and} \quad \vec{b} = \mathbf{G}\vec{c}$, or equivalently

$$\begin{pmatrix} \mathbf{I} & \mathbf{0} & -\mathbf{F} \end{pmatrix} \begin{pmatrix} \vec{a} \\ \vec{b} \\ \vec{c} \end{pmatrix} = \mathbf{0}, \qquad \begin{pmatrix} \mathbf{0} & \mathbf{I} & -\mathbf{G} \end{pmatrix} \begin{pmatrix} \vec{a} \\ \vec{b} \\ \vec{c} \end{pmatrix} = \mathbf{0}$$

Number of linear constraints: $2 \times \sharp$mult. gates

"Compressed" Linear Algebra

Inner Product

- Define $\mathbf{W} = \begin{pmatrix} \mathbf{I} & \mathbf{0} & -\mathbf{F} \\ \mathbf{0} & \mathbf{I} & -\mathbf{G} \end{pmatrix}$, check that $\mathbf{W} \cdot (\vec{a}, \vec{b}, \vec{c}) = \vec{0}$

[2] Aurora's Univariate Sumcheck: Completeness, soundness, efficiency: $\mathscr{R}$ multiplicative subgroup. This work: New simple proof. Completeness, soundness: $\mathscr{R}$ arbitrary. Efficiency: $\mathscr{R}$ multiplicative.

"Compressed" Linear Algebra

Inner Product

- Define $\mathbf{W} = \begin{pmatrix} \mathbf{I} & \mathbf{0} & -\mathbf{F} \\ \mathbf{0} & \mathbf{I} & -\mathbf{G} \end{pmatrix}$, check that $\mathbf{W} \cdot (\vec{a}, \vec{b}, \vec{c}) = \vec{0}$

Linear Algebra World	Polynomial World
$\vec{y} = (y_1, \ldots, y_m)$	$Y(X) = \vec{y} \cdot \vec{\lambda}(X) = \sum_{i=1}^{m} y_i \lambda_i(X)$

[2] Aurora's Univariate Sumcheck: Completeness, soundness, efficiency: $\mathscr{R}$ multiplicative subgroup. This work: New simple proof. Completeness, soundness: $\mathscr{R}$ arbitrary. Efficiency: $\mathscr{R}$ multiplicative.

“Compressed” Linear Algebra

Inner Product

- Define $\mathbf{W} = \begin{pmatrix} \mathbf{I} & \mathbf{0} & -\mathbf{F} \\ \mathbf{0} & \mathbf{I} & -\mathbf{G} \end{pmatrix}$, check that $\mathbf{W} \cdot (\vec{a}, \vec{b}, \vec{c}) = \vec{0}$

Linear Algebra World	Polynomial World
$\vec{y} = (y_1, \ldots, y_m)$	$Y(X) = \vec{y} \cdot \vec{\lambda}(X) = \sum_{i=1}^{m} y_i \lambda_i(X)$
$\vec{w}_i \cdot (\vec{a}, \vec{b}, \vec{c}) = 0$	$\exists R(X)$, $\deg R(X) \leq m - 2$, s.t. $t(X)$ divides $\vec{w}_i(X) \cdot (a(X), b(X), c(X)) - XR(X)$

[2] Aurora’s Univariate Sumcheck: Completeness, soundness, efficiency: $\mathscr{R}$ multiplicative subgroup. This work: New simple proof. Completeness, soundness: $\mathscr{R}$ arbitrary. Efficiency: $\mathscr{R}$ multiplicative.

"Compressed" Linear Algebra

Inner Product

- Define $\mathbf{W} = \begin{pmatrix} \mathbf{I} & \mathbf{0} & -\mathbf{F} \\ \mathbf{0} & \mathbf{I} & -\mathbf{G} \end{pmatrix}$, check that $\mathbf{W} \cdot (\vec{a}, \vec{b}, \vec{c}) = \vec{0}$

Linear Algebra World	Polynomial World
$\vec{y} = (y_1, \ldots, y_m)$	$Y(X) = \vec{y} \cdot \vec{\lambda}(X) = \sum_{i=1}^{m} y_i \lambda_i(X)$
$\vec{w}_i \cdot (\vec{a}, \vec{b}, \vec{c}) = 0$	$\exists R(X)$, $\deg R(X) \leq m - 2$, s.t. $t(X)$ divides $\vec{w}_i(X) \cdot (a(X), b(X), c(X)) - XR(X)$ [2]

[2] Aurora's Univariate Sumcheck: Completeness, soundness, efficiency: $\mathscr{R}$ multiplicative subgroup. This work: New simple proof. Completeness, soundness: $\mathscr{R}$ arbitrary. Efficiency: $\mathscr{R}$ multiplicative.

"Compressed" Linear Algebra

Inner Product

- Define $\mathbf{W} = \begin{pmatrix} \mathbf{I} & \mathbf{0} & -\mathbf{F} \\ \mathbf{0} & \mathbf{I} & -\mathbf{G} \end{pmatrix}$, check that $\mathbf{W} \cdot (\vec{a}, \vec{b}, \vec{c}) = \vec{0}$

Linear Algebra World	Polynomial World
$\vec{y} = (y_1, \ldots, y_m)$	$Y(X) = \vec{y} \cdot \vec{\lambda}(X) = \sum_{i=1}^{m} y_i \lambda_i(X)$
$\vec{w}_i \cdot (\vec{a}, \vec{b}, \vec{c}) = 0$	$\exists R(X)$, $\deg R(X) \leq m - 2$, s.t. $t(X)$ divides $\vec{w}_i(X) \cdot (a(X), b(X), c(X)) - XR(X)$ [2]

$\{\vec{w}_i(X)\}_{i=1}^{m}$ can be computed by the indexer, but 2m inner products

[2] Aurora's Univariate Sumcheck: Completeness, soundness, efficiency: $\mathscr{R}$ multiplicative subgroup. This work: New simple proof. Completeness, soundness: $\mathscr{R}$ arbitrary. Efficiency: $\mathscr{R}$ multiplicative.

Verifiable Subspace Sampling

Algebraic Intuition

Verifiable Subspace Sampling

Algebraic Intuition

$$\begin{pmatrix} \mathbf{I} & \mathbf{0} & -\mathbf{F} \\ \mathbf{0} & \mathbf{I} & -\mathbf{G} \end{pmatrix} \begin{pmatrix} \vec{a} \\ \vec{b} \\ \vec{c} \end{pmatrix} = \vec{0}$$

Verifiable Subspace Sampling

Algebraic Intuition

$$\begin{pmatrix} \mathbf{I} & \mathbf{0} & -\mathbf{F} \\ \mathbf{0} & \mathbf{I} & -\mathbf{G} \end{pmatrix} \begin{pmatrix} \vec{a} \\ \vec{b} \\ \vec{c} \end{pmatrix} = \vec{0}$$

- Sample a random vector $\vec{d}$ in the rowspace of the matrix

Verifiable Subspace Sampling

Algebraic Intuition

$$\begin{pmatrix} \mathbf{I} & \mathbf{0} & -\mathbf{F} \\ \mathbf{0} & \mathbf{I} & -\mathbf{G} \end{pmatrix} \begin{pmatrix} \vec{a} \\ \vec{b} \\ \vec{c} \end{pmatrix} = \vec{0}$$

- Sample a random vector $\vec{d}$ in the rowspace of the matrix
- Check one inner product $\vec{d} \cdot (\vec{a}, \vec{b}, \vec{c}) = 0$.

Verifiable Subspace Sampling

Algebraic Intuition

$$\vec{d}\left\{\mathsf{Samp}(x)\right.^{\top}\begin{pmatrix}\mathbf{I} & \mathbf{0} & -\mathbf{F}\\ \mathbf{0} & \mathbf{I} & -\mathbf{G}\end{pmatrix}\begin{pmatrix}\vec{a}\\ \vec{b}\\ \vec{c}\end{pmatrix}=\vec{0}$$

Verifiable Subspace Sampling

Algebraic Intuition

$$\vec{d} \left\{ \vec{\mathsf{Samp}}(x)^{\top} \right. \begin{pmatrix} \mathbf{I} & \mathbf{0} & -\mathbf{F} \\ \mathbf{0} & \mathbf{I} & -\mathbf{G} \end{pmatrix} \begin{pmatrix} \vec{a} \\ \vec{b} \\ \vec{c} \end{pmatrix} = \vec{0}$$

- Sample a random vector $\vec{d}$ in the rowspace of the matrix

Verifiable Subspace Sampling

Algebraic Intuition

$$\vec{d}\left\{\vec{\mathrm{Samp}}(x)^{\top}\right. \begin{pmatrix} \mathbf{I} & \mathbf{0} & -\mathbf{F} \\ \mathbf{0} & \mathbf{I} & -\mathbf{G} \end{pmatrix} \begin{pmatrix} \vec{a} \\ \vec{b} \\ \vec{c} \end{pmatrix} = \vec{0}$$

- Sample a random vector $\vec{d}$ in the rowspace of the matrix from the function $\vec{\mathrm{Samp}}(X)$ and some challenge x.

Verifiable Subspace Sampling

Algebraic Intuition

$$\vec{d}\left\{\mathrm{Sam}\vec{\mathrm{p}}(x)\right.^{\top}\begin{pmatrix}\mathbf{I} & \mathbf{0} & -\mathbf{F} \\ \mathbf{0} & \mathbf{I} & -\mathbf{G}\end{pmatrix}\begin{pmatrix}\vec{a} \\ \vec{b} \\ \vec{c}\end{pmatrix} = \vec{0}$$

- Sample a random vector $\vec{d}$ in the rowspace of the matrix from the function $\mathrm{Sam}\vec{\mathrm{p}}(X)$ and some challenge x.
- Check one inner product $\vec{d} \cdot (\vec{a}, \vec{b}, \vec{c}) = 0$.

Verifiable Subspace Sampling

In the polynomial world

$$D(X) \quad = \left(\vec{\mathsf{Samp}}(x)^{\top}\mathbf{W}\right)\vec{\lambda}(X) = \vec{d} \cdot \vec{\lambda}(X)$$

Verifiable Subspace Sampling

In the polynomial world

$$\begin{aligned} D(X) &= \left(\vec{\mathsf{Samp}}(x)^{\top}\mathbf{W}\right)\vec{\lambda}(X) = \vec{d}\cdot\vec{\lambda}(X) \\ &= P(X, x) \end{aligned}$$

Verifiable Subspace Sampling

In the polynomial world

$$\begin{aligned} D(X) &= \left(\vec{\mathsf{Samp}}(x)^{\top}\mathbf{W}\right)\vec{\lambda}(X) = \vec{d}\cdot\vec{\lambda}(X) \\ &= P(X, x) \end{aligned}$$

$$P(X, Y) = \left(\vec{\mathsf{Samp}}(Y)^{\top}\mathbf{W}\right)\vec{\lambda}(X)$$

Verifiable Subspace Sampling

In the polynomial world

$$
\begin{aligned}
D(X) &= \left(\vec{\mathsf{Samp}}(x)^{\top}\mathbf{W}\right)\vec{\lambda}(X) = \vec{d}\cdot\vec{\lambda}(X) \\
&= P(X, x)
\end{aligned}
$$

$$P(X, Y) = \left(\vec{\mathsf{Samp}}(Y)^{\top}\mathbf{W}\right)\vec{\lambda}(X)$$

Who samples x?

Verifiable Subspace Sampling

In the polynomial world

$$\begin{aligned} D(X) &= \left(\vec{\mathsf{Samp}}(x)^{\top}\mathbf{W}\right)\vec{\lambda}(X) = \vec{d}\cdot\vec{\lambda}(X) \\ &= P(X, x) \end{aligned}$$

$$P(X, Y) = \left(\vec{\mathsf{Samp}}(Y)^{\top}\mathbf{W}\right)\vec{\lambda}(X)$$

Who samples x?

- *Indexer*: x needs to be secret, resulting in a quadratic SRS (Groth et al. 18)

Verifiable Subspace Sampling

In the polynomial world

$$\begin{aligned} D(X) &= \left(\vec{\mathsf{Samp}}(x)^\top \mathbf{W}\right)\vec{\lambda}(X) = \vec{d}\cdot\vec{\lambda}(X) \\ &= P(X, x) \end{aligned}$$

$$P(X, Y) = \left(\vec{\mathsf{Samp}}(Y)^\top \mathbf{W}\right)\vec{\lambda}(X)$$

Who samples x?

- *Indexer*: x needs to be secret, resulting in a quadratic SRS (Groth et al. 18)
- *Prover*: yeah, sure...

Verifiable Subspace Sampling

In the polynomial world

$$\begin{aligned} D(X) &= \left(\vec{\mathsf{Samp}}(x)^\top \mathbf{W}\right)\vec{\lambda}(X) = \vec{d}\cdot\vec{\lambda}(X) \\ &= P(X, x) \end{aligned}$$

$$P(X, Y) = \left(\vec{\mathsf{Samp}}(Y)^\top \mathbf{W}\right)\vec{\lambda}(X)$$

Who samples x?

- *Indexer*: x needs to be secret, resulting in a quadratic SRS (Groth et al. 18)
- *Prover*: yeah, sure...
- *Verifier*: Who evaluates $P(X, Y) = \left(\vec{\mathsf{Samp}}(Y)^\top \mathbf{W}\right)\vec{\lambda}(X)$ in $Y = x$?

Verifiable Subspace Sampling

Definition

- *Offline phase:* Indexer outputs polynomials describing matrix $\mathbf{W}$.

Verifiable Subspace Sampling

Definition

- *Offline phase:* Indexer outputs polynomials describing matrix $\mathbf{W}$.
- *Online phase:*

Verifiable Subspace Sampling

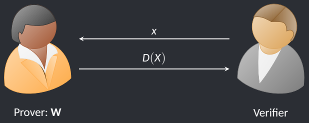
Definition

- *Offline phase:* Indexer outputs polynomials describing matrix $\mathbf{W}$.
- *Online phase:*
  - *Sampling:*

Verifiable Subspace Sampling

Definition

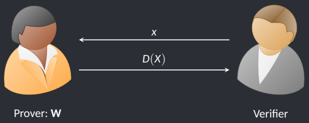
- *Offline phase:* Indexer outputs polynomials describing matrix $\mathbf{W}$.
- *Online phase:*
  - *Sampling:*



Verifiable Subspace Sampling

Definition

- *Offline phase:* Indexer outputs polynomials describing matrix $\mathbf{W}$.
- *Online phase:*
  - *Sampling:*

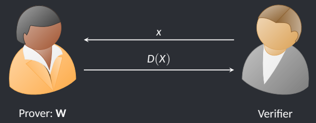


  - *Prove Sampling:*

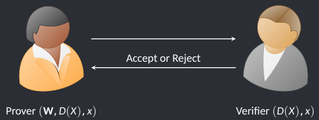
Verifiable Subspace Sampling

Definition

- *Offline phase:* Indexer outputs polynomials describing matrix $\mathbf{W}$.
- *Online phase:*
  - *Sampling:*



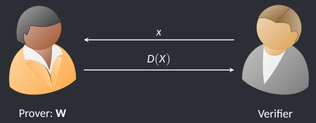
  - *Prove Sampling:*



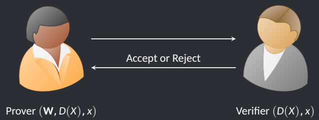
Verifiable Subspace Sampling

Definition

- *Offline phase:* Indexer outputs polynomials describing matrix $\mathbf{W}$.
- *Online phase:*
  - *Sampling:*



  - *Prove Sampling:*



- *Decision phase:* Verifier accepts only if $D(X)$ encodes vector in the rowspace of $\mathbf{W}$ sampled according to x.

Verifiable Subspace Sampling

A common strategy

- Setup: Universal SRS

Verifiable Subspace Sampling

A common strategy

- Setup: Universal SRS
- Preprocessing: on input matrix $\mathbf{W}$, outputs $\mathscr{W}$.

Verifiable Subspace Sampling

A common strategy

- Setup: Universal SRS
- Preprocessing: on input matrix $\mathbf{W}$, outputs $\mathscr{W}$.
- Online phase

Verifiable Subspace Sampling

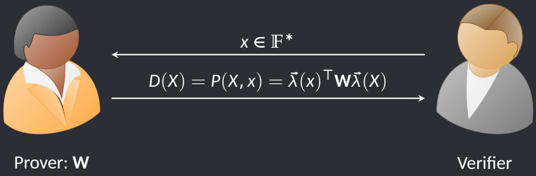
A common strategy

- Setup: Universal SRS
- Preprocessing: on input matrix $\mathbf{W}$, outputs $\mathscr{W}$.
- Online phase
  - Sampling Phase:

Verifiable Subspace Sampling

A common strategy

- Setup: Universal SRS
- Preprocessing: on input matrix $\mathbf{W}$, outputs $\mathscr{W}$.
- Online phase
  - Sampling Phase:



Verifiable Subspace Sampling

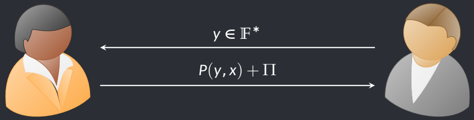
A common strategy

- Prove Sampling Phase:

Verifiable Subspace Sampling

A common strategy

- Prove Sampling Phase:



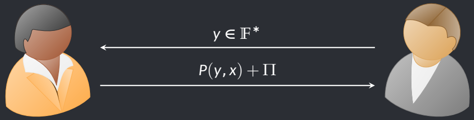
Prover $(\mathbf{W}, D(X), x)$

Verifier $(D(X), x)$, Output: 0/1

Verifiable Subspace Sampling

A common strategy

- Prove Sampling Phase:

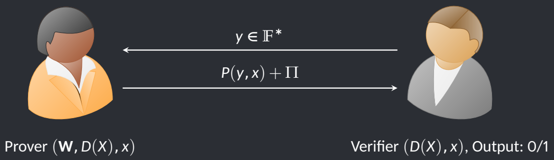


- Decision phase: Verifier accepts only if $D(X)$ encodes $\mathrm{Samp}(\vec{x})^\top \mathbf{W}$.

Verifiable Subspace Sampling

A common strategy

– Prove Sampling Phase:



- Decision phase: Verifier accepts only if $D(X)$ encodes $\text{Samp}(\vec{x})^\top \mathbf{W}$.
- Π is a proof that $P(y, x)$ is correctly evaluated (Signature of Correct Computation of Sonic).

Verifiable Subspace Sampling

State-of-the-art

Verifiable Subspace Sampling

State-of-the-art

- Sonic:

Verifiable Subspace Sampling

State-of-the-art

- Sonic:
  - VSSampling, proof grows with size of decomposition of **W** as permutation.

Verifiable Subspace Sampling

State-of-the-art

- Sonic:
  - VSSampling, proof grows with size of decomposition of $\mathbf{W}$ as permutation.
  - Amortized VSSampling

Verifiable Subspace Sampling

State-of-the-art

- Sonic:
  - VSSampling, proof grows with size of decomposition of $\mathbf{W}$ as permutation.
  - Amortized VSSampling
- Marlin, Lunar:

Verifiable Subspace Sampling

State-of-the-art

- Sonic:
  - VSSampling, proof grows with size of decomposition of $\mathbf{W}$ as permutation.
  - Amortized VSSampling
- Marlin, Lunar:
  - Needs a multiplicative subgroup of size spartsity of matrix, relatively large SRS.

Verifiable Subspace Sampling

State-of-the-art

- Sonic:
  - VSSampling, proof grows with size of decomposition of **W** as permutation.
  - Amortized VSSampling
- Marlin, Lunar:
  - Needs a multiplicative subgroup of size spartsity of matrix, relatively large SRS.
- Our work: (*soon* on eprint)

Verifiable Subspace Sampling

State-of-the-art

- Sonic:
  - VSSampling, proof grows with size of decomposition of $\mathbf{W}$ as permutation.
  - Amortized VSSampling
- Marlin, Lunar:
  - Needs a multiplicative subgroup of size spartsity of matrix, relatively large SRS.
- Our work: (*soon* on eprint)
  - Extended Vandermonde Sampling.

Verifiable Subspace Sampling

State-of-the-art

- Sonic:
  - VSSampling, proof grows with size of decomposition of $\mathbf{W}$ as permutation.
  - Amortized VSSampling
- Marlin, Lunar:
  - Needs a multiplicative subgroup of size spartsity of matrix, relatively large SRS.
- Our work: (*soon* on eprint)
  - Extended Vandermonde Sampling.
  - Reduce SRS size drastically, by decomposing Marlin's VSSampling into simpler building blocks.

Verifiable Subspace Sampling

State-of-the-art

- Sonic:
  - VSSampling, proof grows with size of decomposition of $\mathbf{W}$ as permutation.
  - Amortized VSSampling
- Marlin, Lunar:
  - Needs a multiplicative subgroup of size spartsity of matrix, relatively large SRS.
- Our work: (*soon* on eprint)
  - Extended Vandermonde Sampling.
  - Reduce SRS size drastically, by decomposing Marlin's VSSampling into simpler building blocks.

Verifiable Subspace Sampling

Why?

- Decomposing constructions of universal and updatable SNARKs into blocks that have a well defined algebraic meaning.

Verifiable Subspace Sampling

Why?

- Decomposing constructions of universal and updatable SNARKs into blocks that have a well defined algebraic meaning.
- Captures several constructions.

Verifiable Subspace Sampling

Why?

- Decomposing constructions of universal and updatable SNARKs into blocks that have a well defined algebraic meaning.
- Captures several constructions.
- In fact, VSSampling is the main bottleneck in efficiency/generality in these constructions.

Verifiable Subspace Sampling

Why?

- Decomposing constructions of universal and updatable SNARKs into blocks that have a well defined algebraic meaning.
- Captures several constructions.
- In fact, VSSampling is the main bottleneck in efficiency/generality in these constructions.
- Isolating this component allows to focus on improvements.

Verifiable Subspace Sampling

Why?

- Decomposing constructions of universal and updatable SNARKs into blocks that have a well defined algebraic meaning.
- Captures several constructions.
- In fact, VSSampling is the main bottleneck in efficiency/generality in these constructions.
- Isolating this component allows to focus on improvements.
- Mix and match: we can combine different VSSampling arguments.

Verifiable Subspace Sampling

Why?

- Decomposing constructions of universal and updatable SNARKs into blocks that have a well defined algebraic meaning.
- Captures several constructions.
- In fact, VSSampling is the main bottleneck in efficiency/generality in these constructions.
- Isolating this component allows to focus on improvements.
- Mix and match: we can combine different VSSampling arguments.

Thank you![3]

[3] I am looking for a research stay/internship :) arantxa.zapico@upf.edu