Drinfeld modules in SageMath

`arXiv:2305.00422`

ANTOINE LEUDIÈRE
INRIA Caramba
Joint work with David Ayotte, Xavier Caruso and Yossef Musleh

Journée du DI du Loria

4 July 2023

Outline of the Talk

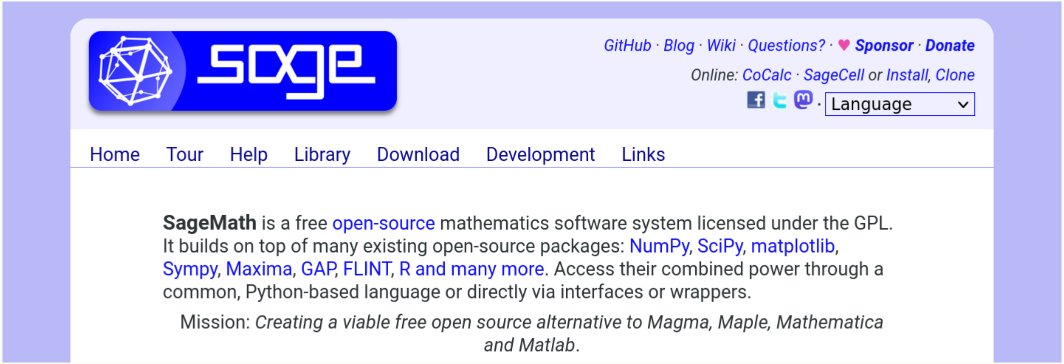
What is SageMath?

What is a Drinfeld module?

What are Drinfeld modules in SageMath?

History

SageMath is the leading computer algebra FOSS system. It was created in 2005 by William Stein; hundreds of mathematicians contributed to it.



Outline of the Talk

What is SageMath?

What is a Drinfeld module?

What are Drinfeld modules in SageMath?

Context

Drinfeld modules were introduced by Vladimir Drinfeld in the 1970s to solve problems from the *class field theory of function fields*.

🤯 Laurent Lafforgue received the Fields medal thanks to Drinfeld modules!

😍 Growing interest for computational research on Drinfeld modules.

😭 Before our contribution, no Drinfeld module implementation in standard systems.

Context

Drinfeld modules were introduced by Vladimir Drinfeld in the 1970s to solve problems from the *class field theory of function fields*.

🤯 Laurent Lafforgue received the Fields medal thanks to Drinfeld modules!

😍 Growing interest for computational research on Drinfeld modules.

😭 Before our contribution, no Drinfeld module implementation in standard systems.

Context

Drinfeld modules were introduced by Vladimir Drinfeld in the 1970s to solve problems from the *class field theory of function fields*.

🤯 Laurent Lafforgue received the Fields medal thanks to Drinfeld modules!

😍 Growing interest for computational research on Drinfeld modules.

😭 Before our contribution, no Drinfeld module implementation in standard systems.

Context

Drinfeld modules were introduced by Vladimir Drinfeld in the 1970s to solve problems from the *class field theory of function fields*.

🤯 Laurent Lafforgue received the Fields medal thanks to Drinfeld modules!

😍 Growing interest for computational research on Drinfeld modules.

😭 Before our contribution, no Drinfeld module implementation in standard systems.

Definition

Let:

- $\mathbb{F}_q$ be a finite field with q elements.
- K be a field containing $\mathbb{F}_q$.
- $\mathbb{F}_q[T]$ be the ring of polynomials with coefficients in $\mathbb{F}_q$.
- $K\{\tau\}$ be the ring of "skew" polynomials $a_0 + a_1\tau + \cdots + a_n\tau^n$ in K satisfying $\tau a_i = a_i^q \tau$, for all $a_i \in K$:

$$K\{\tau\} = \left\{ \sum_{i=0}^{n} a_i \tau^i, \quad n \in \mathbb{Z}_{\geqslant 0}, a_i \in K \right\}.$$

Definition

A Drinfeld module is a special case of $\mathbb{F}_q$-algebra morphism

$$\phi : \mathbb{F}_q[T] \to K\{\tau\}.$$

Definition

Let:

- $\mathbb{F}_q$ be a finite field with q elements.
- K be a field containing $\mathbb{F}_q$.
- $\mathbb{F}_q[T]$ be the ring of polynomials with coefficients in $\mathbb{F}_q$.
- $K\{\tau\}$ be the ring of "skew" polynomials $a_0 + a_1\tau + \cdots + a_n\tau^n$ in K satisfying $\tau a_i = a_i^q \tau$, for all $a_i \in K$:

$$K\{\tau\} = \left\{ \sum_{i=0}^{n} a_i \tau^i, \quad n \in \mathbb{Z}_{\geqslant 0}, a_i \in K \right\}.$$

Definition

A Drinfeld module is a special case of $\mathbb{F}_q$-algebra morphism

$$\phi : \mathbb{F}_q[T] \to K\{\tau\}.$$

Definition

Let:

- $\mathbb{F}_q$ be a finite field with q elements.
- K be a field containing $\mathbb{F}_q$.
- $\mathbb{F}_q[T]$ be the ring of polynomials with coefficients in $\mathbb{F}_q$.
- $K\{\tau\}$ be the ring of "skew" polynomials $a_0 + a_1\tau + \cdots + a_n\tau^n$ in K satisfying $\tau a_i = a_i^q\tau$, for all $a_i \in K$:

$$K\{\tau\} = \left\{\sum_{i=0}^{n} a_i\tau^i, \quad n \in \mathbb{Z}_{\geqslant 0}, a_i \in K\right\}.$$

Definition

A Drinfeld module is a special case of $\mathbb{F}_q$-algebra morphism

$$\phi : \mathbb{F}_q[T] \to K\{\tau\}.$$

Definition

Let:

- $\mathbb{F}_q$ be a finite field with q elements.
- K be a field containing $\mathbb{F}_q$.
- $\mathbb{F}_q[T]$ be the ring of polynomials with coefficients in $\mathbb{F}_q$.
- $K\{\tau\}$ be the ring of "skew" polynomials $a_0 + a_1\tau + \cdots + a_n\tau^n$ in K satisfying $\tau a_i = a_i^q \tau$, for all $a_i \in K$:

$$K\{\tau\} = \left\{ \sum_{i=0}^{n} a_i \tau^i, \quad n \in \mathbb{Z}_{\geqslant 0}, a_i \in K \right\}.$$

Definition

A Drinfeld module is a special case of $\mathbb{F}_q$-algebra morphism

$$\phi : \mathbb{F}_q[T] \to K\{\tau\}.$$

Definition

Let:

- $\mathbb{F}_q$ be a finite field with q elements.
- K be a field containing $\mathbb{F}_q$.
- $\mathbb{F}_q[T]$ be the ring of polynomials with coefficients in $\mathbb{F}_q$.
- $K\{\tau\}$ be the ring of "skew" polynomials $a_0 + a_1\tau + \cdots + a_n\tau^n$ in K satisfying $\tau a_i = a_i^q \tau$, for all $a_i \in K$:

$$K\{\tau\} = \left\{ \sum_{i=0}^{n} a_i \tau^i, \quad n \in \mathbb{Z}_{\geqslant 0}, a_i \in K \right\}.$$

Definition

A Drinfeld module is a special case of $\mathbb{F}_q$-algebra morphism

$$\phi : \mathbb{F}_q[T] \to K\{\tau\}.$$

Definition

Let:

- $\mathbb{F}_q$ be a finite field with q elements.
- K be a field containing $\mathbb{F}_q$.
- $\mathbb{F}_q[T]$ be the ring of polynomials with coefficients in $\mathbb{F}_q$.
- $K\{\tau\}$ be the ring of "skew" polynomials $a_0 + a_1\tau + \cdots + a_n\tau^n$ in K satisfying $\tau a_i = a_i^q \tau$, for all $a_i \in K$:

$$K\{\tau\} = \left\{ \sum_{i=0}^{n} a_i \tau^i, \quad n \in \mathbb{Z}_{\geqslant 0}, a_i \in K \right\}.$$

Definition

A Drinfeld module is a special case of $\mathbb{F}_q$-algebra morphism

$$\phi : \mathbb{F}_q[T] \to K\{\tau\}.$$

Representation

A Drinfeld module $\phi : \mathbb{F}_q[T] \to K\{\tau\}$ can be represented by:

- A morphism.
- A skew polynomial $\phi(T) = g_0 + g_1\tau + \cdots + g_r\tau^r$.
- A list of coefficients $[g_0, g_1, \ldots, g_r]$.

❌ A Drinfeld module is *not* a set!

Representation

A Drinfeld module $\phi : \mathbb{F}_q[T] \to K\{\tau\}$ can be represented by:

- A morphism.
- A skew polynomial $\phi(T) = g_0 + g_1\tau + \cdots + g_r\tau^r$.
- A list of coefficients $[g_0, g_1, \ldots, g_r]$.

❌ A Drinfeld module is *not* a set!

Representation

A Drinfeld module $\phi : \mathbb{F}_q[T] \to K\{\tau\}$ can be represented by:

- A morphism.
- A skew polynomial $\phi(T) = g_0 + g_1\tau + \cdots + g_r\tau^r$.
- A list of coefficients $[g_0, g_1, \ldots, g_r]$.

❌ A Drinfeld module is *not* a set!

Representation

A Drinfeld module $\phi : \mathbb{F}_q[T] \to K\{\tau\}$ can be represented by:

- A morphism.
- A skew polynomial $\phi(T) = g_0 + g_1\tau + \cdots + g_r\tau^r$.
- A list of coefficients $[g_0, g_1, \ldots, g_r]$.

❌ A Drinfeld module is *not* a set!

Representation

A Drinfeld module $\phi : \mathbb{F}_q[T] \to K\{\tau\}$ can be represented by:

- A morphism.
- A skew polynomial $\phi(T) = g_0 + g_1\tau + \cdots + g_r\tau^r$.
- A list of coefficients $[g_0, g_1, \dots, g_r]$.

❌ A Drinfeld module is *not* a set!

Outline of the Talk

What is SageMath?

What is a Drinfeld module?

What are Drinfeld modules in SageMath?

The Parent/Element framework

SageMath is built on the Parent/Element framework: SageMath objects are usually either a set (`Parent`) or an element in the set (`Element`). And `Parents` should belong to a category. This does not really fit Drinfeld modules:

- Drinfeld form a category, and as such should be `Parents`.
- But Drinfeld modules have no underlying sets, so they don't have elements and as such should not be `Parents`.
- A Drinfeld module is a special kind of morphism, so it *technically* is an element in a set of morphisms, but mathematicians do not think about them this way.

The Parent/Element framework

SageMath is built on the Parent/Element framework: SageMath objects are usually either a set (`Parent`) or an element in the set (`Element`). And `Parents` should belong to a category. This does not really fit Drinfeld modules:

- Drinfeld form a category, and as such should be `Parents`.
- But Drinfeld modules have no underlying sets, so they don't have elements and as such should not be `Parents`.
- A Drinfeld module is a special kind of morphism, so it *technically* is an element in a set of morphisms, but mathematicians do not think about them this way.

The Parent/Element framework

SageMath is built on the Parent/Element framework: SageMath objects are usually either a set (`Parent`) or an element in the set (`Element`). And `Parents` should belong to a category. This does not really fit Drinfeld modules:

- Drinfeld form a category, and as such should be `Parents`.
- But Drinfeld modules have no underlying sets, so they don't have elements and as such should not be `Parents`.
- A Drinfeld module is a special kind of morphism, so it *technically* is an element in a set of morphisms, but mathematicians do not think about them this way.

The Parent/Element framework

SageMath is built on the Parent/Element framework: SageMath objects are usually either a set (`Parent`) or an element in the set (`Element`). And `Parents` should belong to a category. This does not really fit Drinfeld modules:

- Drinfeld form a category, and as such should be `Parents`.
- But Drinfeld modules have no underlying sets, so they don't have elements and as such should not be `Parents`.
- A Drinfeld module is a special kind of morphism, so it *technically* is an element in a set of morphisms, but mathematicians do not think about them this way.

Possible solutions

There are multiple possible solutions:

1. Making Drinfeld modules `Parents` without `Elements`. In fact this solution has a strong mathematical soundness. Drawbacks: `Parents` are supposed to have elements; their category must be a subcategory of the category of sets.
2. Making Drinfeld modules a `CategoryObject`. Drawbacks: this class is barely used in the codebase.
3. Making Drinfeld modules elements and their category a Parent without a category. Drawbacks: no mathematical satisfaction, and this prevents from having a standard implementation for morphisms.

After a passionate debate with the community, we chose to make Drinfeld modules `Parents` without `Elements`.

Possible solutions

There are multiple possible solutions:

1. Making Drinfeld modules `Parents` without `Elements`. In fact this solution has a strong mathematical soundness. Drawbacks: `Parents` are supposed to have elements; their category must be a subcategory of the category of sets.
2. Making Drinfeld modules a `CategoryObject`. Drawbacks: this class is barely used in the codebase.
3. Making Drinfeld modules elements and their category a Parent without a category. Drawbacks: no mathematical satisfaction, and this prevents from having a standard implementation for morphisms.

After a passionate debate with the community, we chose to make Drinfeld modules `Parents` without `Elements`.

Possible solutions

There are multiple possible solutions:

1. Making Drinfeld modules `Parents` without `Elements`. In fact this solution has a strong mathematical soundness. Drawbacks: `Parents` are supposed to have elements; their category must be a subcategory of the category of sets.
2. Making Drinfeld modules a `CategoryObject`. Drawbacks: this class is barely used in the codebase.
3. Making Drinfeld modules elements and their category a Parent without a category. Drawbacks: no mathematical satisfaction, and this prevents from having a standard implementation for morphisms.

After a passionate debate with the community, we chose to make Drinfeld modules `Parents` without `Elements`.

Possible solutions

There are multiple possible solutions:

1. Making Drinfeld modules `Parents` without `Elements`. In fact this solution has a strong mathematical soundness. Drawbacks: `Parents` are supposed to have elements; their category must be a subcategory of the category of sets.
2. Making Drinfeld modules a `CategoryObject`. Drawbacks: this class is barely used in the codebase.
3. Making Drinfeld modules elements and their category a Parent without a category. Drawbacks: no mathematical satisfaction, and this prevents from having a standard implementation for morphisms.

After a passionate debate with the community, we chose to make Drinfeld modules `Parents` without `Elements`.

Possible solutions

There are multiple possible solutions:

1. Making Drinfeld modules `Parents` without `Elements`. In fact this solution has a strong mathematical soundness. Drawbacks: `Parents` are supposed to have elements; their category must be a subcategory of the category of sets.
2. Making Drinfeld modules a `CategoryObject`. Drawbacks: this class is barely used in the codebase.
3. Making Drinfeld modules elements and their category a Parent without a category. Drawbacks: no mathematical satisfaction, and this prevents from having a standard implementation for morphisms.

After a passionate debate with the community, we chose to make Drinfeld modules `Parents` without `Elements`.

Conclusion

- Drinfeld modules are in SageMath! Generalist implementation with comprehensive documentation.
- We received lots of positive feedback and new features are being actively implemented and reviewed.
- We got a software presentation accepted at ISSAC '23: `arXiv:2305.00422`.

Conclusion

- Drinfeld modules are in SageMath! Generalist implementation with comprehensive documentation.
- We received lots of positive feedback and new features are being actively implemented and reviewed.
- We got a software presentation accepted at ISSAC '23: `arXiv:2305.00422`.

Conclusion

- Drinfeld modules are in SageMath! Generalist implementation with comprehensive documentation.
- We received lots of positive feedback and new features are being actively implemented and reviewed.
- We got a software presentation accepted at ISSAC '23: `arXiv:2305.00422`.