Solving in Computer Algebra

Prof. Dr. Wolfram Koepf

Universität Kassel

http://www.mathematik.uni-kassel.de/~koepf

October 9, 2017, 9:30
Douala, Cameroon

Abstract

Topics of this lecture

Abstract

Topics of this lecture

- In this lecture, we will discuss how Maxima can solve many mathematical questions.

Abstract

Topics of this lecture

- In this lecture, we will discuss how Maxima can solve many mathematical questions.
- In several cases these questions will be considered in much more detail by other plenary speakers.

Abstract

Topics of this lecture

- In this lecture, we will discuss how Maxima can solve many mathematical questions.
- In several cases these questions will be considered in much more detail by other plenary speakers.
- The topics considered include the following:

Abstract

Topics of this lecture

- In this lecture, we will discuss how Maxima can solve many mathematical questions.
- In several cases these questions will be considered in much more detail by other plenary speakers.
- The topics considered include the following:
  - Solving linear systems

Abstract

Topics of this lecture

- In this lecture, we will discuss how Maxima can solve many mathematical questions.
- In several cases these questions will be considered in much more detail by other plenary speakers.
- The topics considered include the following:
  - Solving linear systems
  - Solving polynomial equations

Abstract

Topics of this lecture

- In this lecture, we will discuss how Maxima can solve many mathematical questions.
- In several cases these questions will be considered in much more detail by other plenary speakers.
- The topics considered include the following:
  - Solving linear systems
  - Solving polynomial equations
  - Solving polynomial systems

Abstract

Topics of this lecture

- In this lecture, we will discuss how Maxima can solve many mathematical questions.
- In several cases these questions will be considered in much more detail by other plenary speakers.
- The topics considered include the following:
  - Solving linear systems
  - Solving polynomial equations
  - Solving polynomial systems
  - Solving modular equations: Chinese remainder theorem, discrete logarithms

Abstract

Topics of this lecture

- In this lecture, we will discuss how Maxima can solve many mathematical questions.
- In several cases these questions will be considered in much more detail by other plenary speakers.
- The topics considered include the following:
  - Solving linear systems
  - Solving polynomial equations
  - Solving polynomial systems
  - Solving modular equations: Chinese remainder theorem, discrete logarithms
  - Solving differential equations

Abstract

Topics of this lecture

- In this lecture, we will discuss how Maxima can solve many mathematical questions.
- In several cases these questions will be considered in much more detail by other plenary speakers.
- The topics considered include the following:
  - Solving linear systems
  - Solving polynomial equations
  - Solving polynomial systems
  - Solving modular equations: Chinese remainder theorem, discrete logarithms
  - Solving differential equations
  - Solving recurrence equations

Solving linear systems

Gaussian elimination

Solving linear systems

Gaussian elimination

- For example using Gaussian elimination every linear system of equations can be solved algorithmically.

Solving linear systems

Gaussian elimination

- For example using Gaussian elimination every linear system of equations can be solved algorithmically.
- In Maxima such systems can be solved using the solve command.

Solving linear systems

Gaussian elimination

- For example using Gaussian elimination every linear system of equations can be solved algorithmically.
- In Maxima such systems can be solved using the solve command.
- Start Maxima

Solving linear systems

Gaussian elimination

- For example using Gaussian elimination every linear system of equations can be solved algorithmically.
- In Maxima such systems can be solved using the solve command.
- Start Maxima
- solve is a general solver which scans the input and tries to find the appropriate algorithm.

Solving linear systems

Gaussian elimination

- For example using Gaussian elimination every linear system of equations can be solved algorithmically.
- In Maxima such systems can be solved using the solve command.
- Start Maxima
- solve is a general solver which scans the input and tries to find the appropriate algorithm.
- The specific solver for linear systems is called linsolve.

Solving linear systems

Gaussian elimination

- For example using Gaussian elimination every linear system of equations can be solved algorithmically.
- In Maxima such systems can be solved using the solve command.
- Start Maxima
- solve is a general solver which scans the input and tries to find the appropriate algorithm.
- The specific solver for linear systems is called linsolve.
- We saw already last week that one can also work with matrices and compute determinants etc.

Solving linear systems

Gaussian elimination

- For example using Gaussian elimination every linear system of equations can be solved algorithmically.
- In Maxima such systems can be solved using the solve command.
- Start Maxima
- solve is a general solver which scans the input and tries to find the appropriate algorithm.
- The specific solver for linear systems is called linsolve.
- We saw already last week that one can also work with matrices and compute determinants etc.
- Martin Kreuzer will give more details about modern linear algebra techniques in his talk.

Polynomial Equations

Cardano and Ferrari

Polynomial Equations

Cardano and Ferrari

- As we saw already last week, polynomial equations can be solved up to degree 4 symbolically.

Polynomial Equations

Cardano and Ferrari

- As we saw already last week, polynomial equations can be solved up to degree 4 symbolically.
- For this purpose, formulas by Cardano and Ferrari (1545) are used, see https://en.wikipedia.org/wiki/Cubic_function and https://en.wikipedia.org/wiki/Quartic_function.

Polynomial Equations

Cardano and Ferrari

- As we saw already last week, polynomial equations can be solved up to degree 4 symbolically.
- For this purpose, formulas by Cardano and Ferrari (1545) are used, see https://en.wikipedia.org/wiki/Cubic_function and https://en.wikipedia.org/wiki/Quartic_function.

Polynomial factorization

- In algebra it is proved that, in the generic case, the zeros of a polynomial of degree larger than 4 cannot be given by a formula.

Polynomial Equations

Cardano and Ferrari

- As we saw already last week, polynomial equations can be solved up to degree 4 symbolically.
- For this purpose, formulas by Cardano and Ferrari (1545) are used, see https://en.wikipedia.org/wiki/Cubic_function and https://en.wikipedia.org/wiki/Quartic_function.

Polynomial factorization

- In algebra it is proved that, in the generic case, the zeros of a polynomial of degree larger than 4 cannot be given by a formula.
- Nevertheless in special cases, polynomial zeros can be computed. For this purpose, polynomial factorization (over $\mathbb{Q}$) is used. This method is accessible via the factor and solve commands.

Polynomial Systems

Varieties and polynomial ideals

Polynomial Systems

Varieties and polynomial ideals

- The zero set of a multivariate polynomial system is called a variety. The variety is related to the corresponding polynomial ideal generated by the polynomials.

Polynomial Systems

Varieties and polynomial ideals

- The zero set of a multivariate polynomial system is called a variety. The variety is related to the corresponding polynomial ideal generated by the polynomials.
- Classical methods to solve polynomial systems of equations use either resultants or Groebner bases.

Polynomial Systems

Varieties and polynomial ideals

- The zero set of a multivariate polynomial system is called a variety. The variety is related to the corresponding polynomial ideal generated by the polynomials.
- Classical methods to solve polynomial systems of equations use either resultants or Groebner bases.

Resultants

- Resultants can be used to eliminate variables in polynomial systems.

Polynomial Systems

Varieties and polynomial ideals

- The zero set of a multivariate polynomial system is called a variety. The variety is related to the corresponding polynomial ideal generated by the polynomials.
- Classical methods to solve polynomial systems of equations use either resultants or Groebner bases.

Resultants

- Resultants can be used to eliminate variables in polynomial systems.
- Using this method, we solve the system

$$\begin{aligned} x + y + z &= 6 \\ x^2 + y^2 + z^2 &= 14 \\ x^4 + y^4 + z^4 &= 98\,. \end{aligned}$$

Chinese Remainder Theorem

Chinese remainder theorem

Chinese Remainder Theorem

Chinese remainder theorem

- Several equations of the type

$$\begin{aligned} x &\equiv a_1 \pmod{p_1} \\ x &\equiv a_2 \pmod{p_2} \\ &\vdots \\ x &\equiv a_n \pmod{p_n}, \end{aligned}$$

  can be solved using the Chinese remainder theorem.

Chinese Remainder Theorem

Chinese remainder theorem

- Several equations of the type

$$\begin{aligned} x &\equiv a_1 \pmod{p_1} \\ x &\equiv a_2 \pmod{p_2} \\ &\vdots \\ x &\equiv a_n \pmod{p_n}\,, \end{aligned}$$

  can be solved using the Chinese remainder theorem.
- For the solution the extended Euclidean algorithm is used.

Chinese Remainder Theorem

Chinese remainder theorem

- Several equations of the type

$$\begin{aligned} x &\equiv a_1 \pmod{p_1} \\ x &\equiv a_2 \pmod{p_2} \\ &\vdots \\ x &\equiv a_n \pmod{p_n}\,, \end{aligned}$$

  can be solved using the Chinese remainder theorem.
- For the solution the extended Euclidean algorithm is used.
- As an example, we solve the system

$$\begin{aligned} x &\equiv 0 \pmod{2} \\ x &\equiv 0 \pmod{3} \\ x &\equiv 1 \pmod{7}\,, \end{aligned}$$

Discrete Logarithms

Discrete logarithms

Discrete Logarithms

Discrete logarithms

- If we want to solve the equation

$$a^x \equiv b \pmod{p},$$

for the exponent x, then such a solution is called the discrete logarithm of b w. r. t. base a.

Discrete Logarithms

Discrete logarithms

- If we want to solve the equation

$$a^x \equiv b \pmod{p},$$

for the exponent x, then such a solution is called the discrete logarithm of b w. r. t. base a.

- The discrete logarithm is the inverse function of the modular exponential function $a^x \pmod{p}$.

Discrete Logarithms

Discrete logarithms

- If we want to solve the equation

$$a^x \equiv b \pmod{p},$$

  for the exponent x, then such a solution is called the discrete logarithm of b w. r. t. base a.
- The discrete logarithm is the inverse function of the modular exponential function $a^x \pmod{p}$.
- It turns out that no efficient algorithm is known to compute the discrete logarithm!

Discrete Logarithms

Discrete logarithms

- If we want to solve the equation

$$a^x \equiv b \pmod{p},$$

  for the exponent x, then such a solution is called the discrete logarithm of b w. r. t. base a.
- The discrete logarithm is the inverse function of the modular exponential function $a^x \pmod{p}$.
- It turns out that no efficient algorithm is known to compute the discrete logarithm!
- Why is it so difficult to compute discrete logarithms?

Discrete Logarithms

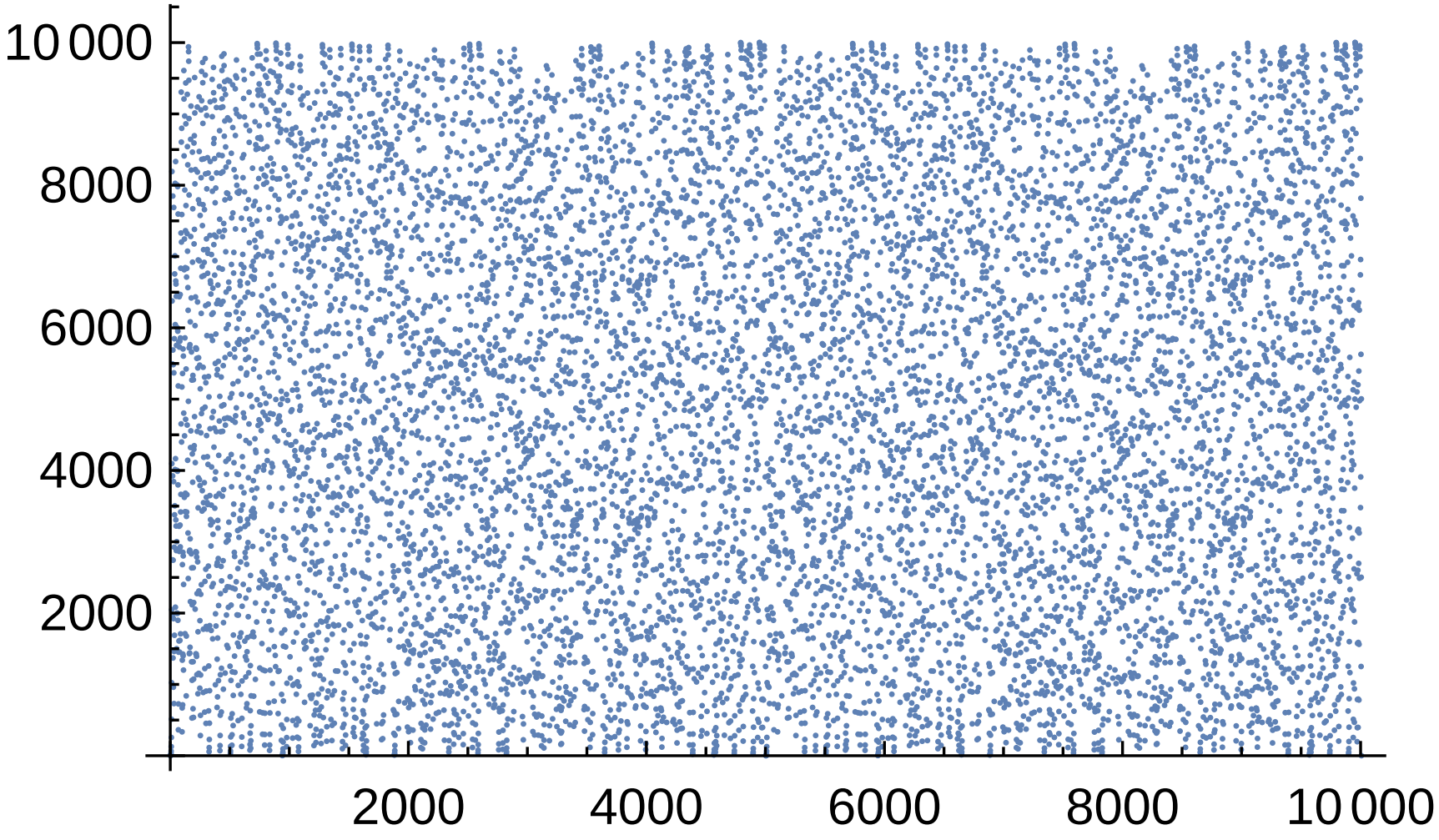
Discrete logarithms

- If we want to solve the equation

$$a^x \equiv b \pmod{p},$$

  for the exponent x, then such a solution is called the discrete logarithm of b w. r. t. base a.
- The discrete logarithm is the inverse function of the modular exponential function $a^x \pmod{p}$.
- It turns out that no efficient algorithm is known to compute the discrete logarithm!
- Why is it so difficult to compute discrete logarithms?
- To visualize the main reason for this difficulty, we plot the modular exponential function.

Discrete Logarithms



Differential Equations

First order ODEs

Differential Equations

First order ODEs

- Solution methods for first order differential equations are known if they are

Differential Equations

First order ODEs

- Solution methods for first order differential equations are known if they are
  - linear,

Differential Equations

First order ODEs

- Solution methods for first order differential equations are known if they are
  - linear,
  - separable,

Differential Equations

First order ODEs

- Solution methods for first order differential equations are known if they are
  - linear,
  - separable,
  - exact,

Differential Equations

First order ODEs

- Solution methods for first order differential equations are known if they are
  - linear,
  - separable,
  - exact,
  - Bernoulli.

Differential Equations

First order ODEs

- Solution methods for first order differential equations are known if they are
  - linear,
  - separable,
  - exact,
  - Bernoulli.
- The Maxima command ode2 applies the above methods.

Differential Equations

First order ODEs

- Solution methods for first order differential equations are known if they are
  - linear,
  - separable,
  - exact,
  - Bernoulli.
- The Maxima command ode2 applies the above methods.

Higher order ODEs

- . . . are solved if they are linear with constant coefficients.

Differential Equations

First order ODEs

- Solution methods for first order differential equations are known if they are
  - linear,
  - separable,
  - exact,
  - Bernoulli.
- The Maxima command ode2 applies the above methods.

Higher order ODEs

- . . . are solved if they are linear with constant coefficients.
- The command desolve uses the Laplace transform.

Differential Equations

First order ODEs

- Solution methods for first order differential equations are known if they are
  - linear,
  - separable,
  - exact,
  - Bernoulli.
- The Maxima command ode2 applies the above methods.

Higher order ODEs

- . . . are solved if they are linear with constant coefficients.
- The command desolve uses the Laplace transform.
- The package contrib_ode can detect special function solutions.

Differential Equations

First order ODEs

- Solution methods for first order differential equations are known if they are
  - linear,
  - separable,
  - exact,
  - Bernoulli.
- The Maxima command ode2 applies the above methods.

Higher order ODEs

- . . . are solved if they are linear with constant coefficients.
- The command desolve uses the Laplace transform.
- The package contrib_ode can detect special function solutions. *Mathematica* and *Maple* know even more.

Differential Equations

First order ODEs

- Solution methods for first order differential equations are known if they are
  - linear,
  - separable,
  - exact,
  - Bernoulli.
- The Maxima command ode2 applies the above methods.

Higher order ODEs

- . . . are solved if they are linear with constant coefficients.
- The command desolve uses the Laplace transform.
- The package contrib_ode can detect special function solutions. *Mathematica* and *Maple* know even more.
- Georg Regensburger will give more details in his talks.

Recurrence Equations

Holonomic first order recurrences

Recurrence Equations

Holonomic first order recurrences

- Every linear first order recurrence with polynomial coefficients

$$(k + b_1) \cdots (k + b_q) a_{k+1} = (k + a_1) \cdots (k + a_p)\, x\, a_k$$

Recurrence Equations

Holonomic first order recurrences

- Every linear first order recurrence with polynomial coefficients

$$(k+b_1)\cdots(k+b_q)a_{k+1} = (k+a_1)\cdots(k+a_p)\,x\,a_k$$

can be easily solved as

$$a_k = a_0 \frac{(a_1)_k \ldots (a_p)_k}{(b_1)_k \cdots (b_q)_k}\, x^k$$

Recurrence Equations

Holonomic first order recurrences

- Every linear first order recurrence with polynomial coefficients

$$(k+b_1)\cdots(k+b_q)a_{k+1} = (k+a_1)\cdots(k+a_p)\,x\,a_k$$

can be easily solved as

$$a_k = a_0 \frac{(a_1)_k \ldots (a_p)_k}{(b_1)_k \cdots (b_q)_k}\, x^k$$

in terms of the Pochhammer symbol (shifted factorial)

$$(a)_k = a(a+1)\cdots(a+k-1) = \frac{\Gamma(a+k)}{\Gamma(a)}\ .$$

Recurrence Equations

Holonomic first order recurrences

- Every linear first order recurrence with polynomial coefficients

$$(k+b_1)\cdots(k+b_q)a_{k+1} = (k+a_1)\cdots(k+a_p)\,x\,a_k$$

can be easily solved as

$$a_k = a_0 \frac{(a_1)_k \ldots (a_p)_k}{(b_1)_k \cdots (b_q)_k}\, x^k$$

in terms of the Pochhammer symbol (shifted factorial)

$$(a)_k = a(a+1)\cdots(a+k-1) = \frac{\Gamma(a+k)}{\Gamma(a)}\ .$$

- The Maxima package solve_rec finds these solutions.

Recurrence Equations

Holonomic first order recurrences

- Every linear first order recurrence with polynomial coefficients

$$(k+b_1)\cdots(k+b_q)a_{k+1} = (k+a_1)\cdots(k+a_p)\,x\,a_k$$

can be easily solved as

$$a_k = a_0 \frac{(a_1)_k \ldots (a_p)_k}{(b_1)_k \cdots (b_q)_k}\, x^k$$

in terms of the Pochhammer symbol (shifted factorial)

$$(a)_k = a(a+1)\cdots(a+k-1) = \frac{\Gamma(a+k)}{\Gamma(a)}\ .$$

- The Maxima package solve_rec finds these solutions.
- However, the general solutions contain the Gamma function $\Gamma(k+1) = k!$.

Recurrence Equations

Holonomic first order recurrences

- Every linear first order recurrence with polynomial coefficients

$$(k+b_1)\cdots(k+b_q)a_{k+1} = (k+a_1)\cdots(k+a_p)\,x\,a_k$$

can be easily solved as

$$a_k = a_0 \frac{(a_1)_k \ldots (a_p)_k}{(b_1)_k \cdots (b_q)_k}\, x^k$$

in terms of the Pochhammer symbol (shifted factorial)

$$(a)_k = a(a+1)\cdots(a+k-1) = \frac{\Gamma(a+k)}{\Gamma(a)}\ .$$

- The Maxima package solve_rec finds these solutions.
- However, the general solutions contain the Gamma function $\Gamma(k+1) = k!$.
- More details will be given in my talk next Friday.

Many thanks for your interest!