Some Results on Algebraic Cryptanalysis of A5/2 Algorithm

Adela MIHAITA

Faculty of Mathematics and Computer Science, University of Bucharest
Bucharest, ROMANIA

Abstract: Algebraic cryptanalysis of A5/2 cipher is the is the subject of this article. Recovering the secret key is reduced to solving a system with several hundreds of quadratic equations for which various methods have been proposed during years. This paper presents some experimental results (concerning the time and the number of frames needed) using different implementations of Groebner basis algorithm for solving the system.

Keywords: cryptanalysis, A5

1. Introduction

The Global System for Mobile Communications (GSM) is the most widely used cellular technology in the world. The most important threat against communication systems is eavesdropping on conversations. Its confidentiality resides in two A5 series algorithms, the _rst of which is the most widespread in Europe and USA, and the second on the Asian continent.
There are some impressive cryptanalytic attacks which manage to make possible real-time listening of conversations. Although many of them are based on a significant hardware power and on unrealistic requirements, due to technical progress, the latest attacks tend to become more and more accessible (most studies have been done on algorithm A5 / 1 because its area of coverage includes the most powerful countries of the world). Still, it seems that their implementation is problematic. The attacks exploit aws in the system, such as the fact that GSM employs error correction before encryption in the transmission path.
In this paper, we are interested in A5/2 stream cipher, a cipher used to provide voice privacy in GSM. It was developed for countries outside Europe and U.S. but is simpler and weaker than A5/1. The purpose of this paper is an algebraic analysis of the cipher A5/2 and a presentation of practical results obtained in the study of its cryptanalysis.

This is a post conference paper. Parts of this paper have been published in the Proceedings of the SECITC 2009 Conference (printed version).

2. Brief presentation of A5/2 algorithm

It is worth noting that A5/2 is built on top of A5/1s architecture.
The stream cipher A5/2 accepts a 64-bit key K_c, and a 22-bit publicly known initial value called Count (which is derived from the publicly known frame number). In GSM, communication is done by frames sites where a frame is transmitted every 4.6 milliseconds. In each frame, A5 / 2 is initialized with the session key and the frame number. The stream key obtained at the output has 228 bits and is divided into two halves: first half (114 bits) is used to encrypt data transmitted from the mobile phone to network while the second half is used to encrypt the link from the mobile phone to network.
Encryption is performed as a bitwise XOR of the plaintext message with the keystream generated by the A5/2 cipher. Should be noted that the base station applies the same algorithm A5/2, so that each of the two parties will communicate using the first half of the key generated to encrypt the data sent and the second half to decrypt (decryption is done similar to encryption, by applying XOR operation) the data received.
The internal state of A5/2 is composed of four Linear Feedback Shift Registers (LFSRs): R1, R2, R3, and R4, of lengths

ISSN 2067 – 4074

19-bit, 22-bit, 23-bit, and 17-bit, respectively, with linear feedback as shown in the figure. The basic operation of each register is clocking (regular), through which the feedback is calculated (XOR 1 between the positions of the register highlighted in the figure), then the register is shifted a bit to the right (removing the rightmost bit which becomes output bit) and the feedback calculated is stored in the leftmost position of the register.
The clocking operation is (regular) (as described above) during initialization of the internal state with K_c and f and (irregular) during key generation, as will be described in detail later in this paper.
A5/2 is initialized with K_c and f in a four-step key setup, as described below, in the key setup, where the i'th bit of Kc is denoted by Kc[i], the i'th bit of f is denoted by f[i], and i = 0 is the least significant bit.

1. Set R1 = R2 = R3 = R4 = 0.
2. For i = 0 to 63
   - Clock all four registers
   - $R1[0] \leftarrow R1[0] \oplus K_c[i];\ R2[0] \leftarrow R2[0] \oplus K_c[i];\ R3[0] \leftarrow R3[0] \oplus K_c[i];\ R4[0] \leftarrow R4[0] \oplus K_c[i].$
3. For i = 0 to 21
   - Clock all four registers
   - $R1[0] \leftarrow R1[0] \oplus f[i];\ R2[0] \leftarrow R2[0] \oplus f[i];\ R3[0] \leftarrow R3[0] \oplus f[i];\ R4[0] \leftarrow R4[0] \oplus f[i].$
4. Set the bits $R1[15] \leftarrow 1, R2[16] \leftarrow 1, R3[18] \leftarrow 1, R4[10] \leftarrow 1$

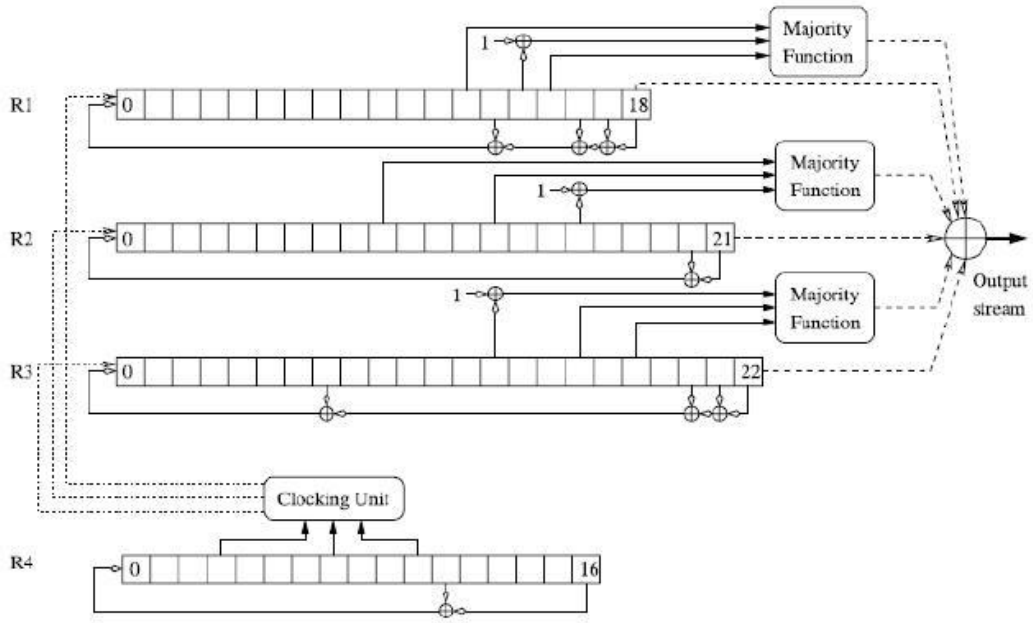

Fig. 1. Internal structure of A5/2 cipher

A5/2 works in cycles, where at the end of each cycle one output bit is produced. During each cycle two or three of registers R1, R2, and R3 are clocked by a clocking unit, based on the value of three bits of R4: R4[3], R4[7], and R4[10]. The clocking unit performs a majority function on the bits : maj(a; b; c) = ab ⊕ ac ⊕ bc where a; b; c are three bits from R4. Then, the registers are clocked as follows: R1 is clocked if and only if R4[10] agrees with the majority. R2 is clocked if and only if R4[3] agrees with the majority. R3 is clocked if and only if R4[7] agrees with the majority. After these clockings, R4 is clocked, and an output bit is generated from the values of R1, R2, and R3, by XOR-ing their rightmost bits to three majority values, one of each register.
Note that the majority function (used for the output) is quadratic in its input.
The first 99 output bits are discarded, and the next 228 output bits form the keystream needed for encryption.

Generation process can be summarized as follows:
1. Run the key setup with K_c and f
2. Run A5/2 for 99 de cycles and discard the output
3. Run A5/2 for 228 cycles and use the output as keystream

3. Algebraic cryptanalysis

Algebraic attacks form an important part of cryptanalysis techniques. They address both (stream ciphers) like A5 series algorithms, Toyocrypt, Bluetooth key stream generator but also block ciphers (AES, Serpent). The efficiency of such procedures depends on the efficiency of the algorithm used to generate algebraic equations and solve a large system of multivariated equations.
Generation of equations is cipher specific, while solving the system of multivariate algebraic equations
is a much studied problem in the field of computational algebraic geometry and commutative algebra. This problem is NP-complete even if all the equations are quadratic and the field is GF(2). The simplest method for solving systems of multivariate polynomial equations is (linearization) (adding new variables that correspond to all monomials present in equations). After that, methods much better and more efficient were proposed: reliniarization [3] as method for solving systems with $\in$n2 quadratic equations in n variables with epsilon < 1/2. The basic idea is to add to the linear system obtained by linearization, equations expressing the fact that additional variables introduced by linearization are related rather than independent. Reliniarization solves many systems of equations that linearization does not, yet the complexity and success rate are not well understood. XL method (eXtended Linearization) can be viewed as a combination between linearization method and Groebner bases. The idea is to generate, from each polynomial equation, a large number of highly degree variants by multiplying the equation with all possible monomials not exceed a certain degree, and then linearization of the extended system. This method was later improved to other algorithms. The concept of Groebner bases was introduced in 1965 as Buchberger's algorithm which is the classical algorithm for calculating Groebner bases. Very fast implementations of the algorithm (F4 and F5) have proposed it as a suitable technique for algebraic attacks. Groebner bases is a general method for solving systems of polynomial equations. The efficiency of such procedures depends essentially on the algorithm used for calculating Groebner bases.

4. The system of algebraic polynomial equations for the A5/2 cipher

The entire A5/2 cipher can be expressed as a very large system of multivariate algebraic equations that can be solved in order to recover the secret key.
From the structure of the stream key generator of A5/2, we deduce the following equation for an output bit at moment t, where x_0, … x_{18}, y_0, … y_{21}, z_0, … z_{22} and A_0, …A_{16} represent the internal state of R1, R2, R3 and R4, respectively.

$$output-bit^t = x_{18}^t \oplus maj(x_{12}^t, 1+x_{14}^t, x_{15}^t) \oplus y_{21}^t \oplus maj(y_9^t, y_{13}^t, 1 \oplus y_{16}^t) \oplus z_{22}^{\parallel} \oplus maj(1 \oplus z_{13}^t, z_{16}^t, z_{18}^t)$$

Also, there can be defined algebraic relationships between bits of the register R1, based on the clocking procedure described above:

$$x_p^t = x_p^{t-1} \cdot (A_{10}^{t-1} \oplus maj(A_3^{t-1}, A_7^{t-1}, A_{10}^{t-1})) \oplus x_{p-1}^{t-1}(1 \oplus A_{10}^{t-1} \oplus maj(A_3^{t-1}, A_7^{t-1}, A_{10}^{t-1})) \text{ for } p = 1 \cdots 18$$

and:

$$x_0^t = x_0^{t-1} \cdot (A_{10}^{t-1} \oplus maj(A_3^{t-1}, A_7^{t-1}, A_{10}^{t-1})) \oplus feedbackR_1^{t-1}(1 \oplus A_{10}^{t-1} \oplus maj(A_3^{t-1}, A_7^{t-1}, A_{10}^{t-1}))$$

ISSN 2067 – 4074

where the feedback function for R1 is defined as it follows: $feedbackR_1^{t-1} = x_{13}^{t-1} \oplus x_{16}^{t-1} \oplus x_{17}^{t-1} \oplus x_{18}^{t-1}$

Similarly, the register R2 can define these relationships among bits:

$$y_q^t = y_q^{t-1} \cdot (A_3^{t-1} \oplus maj(A_3^{t-1}, A_7^{t-1}, A_{10}^{t-1})) \oplus y_{q-1}^{t-1}(1 \oplus A_3^{t-1} \oplus maj(A_3^{t-1}, A_7^{t-1}, A_{10}^{t-1})), \text{ for } q = 1 \cdots 21$$

and:

$$y_0^t = y_0^{t-1} \cdot (A_3^{t-1} \oplus maj(A_3^{t-1}, A_7^{t-1}, A_{10}^{t-1})) \oplus feedbackR_2^{t-1}(1 \oplus A_3^{t-1} \oplus maj(A_3^{t-1}, A_7^{t-1}, A_{10}^{t-1}))$$

where the feedback function for R2 is defined as it follows: $feedbackR_2^{t-1} = y_{21}^{t-1} \oplus y_{22}^{t-1}$

$$z_r^t = z_r^{t-1} \cdot (A_7^{t-1} \oplus maj(A_3^{t-1}, A_7^{t-1}, A_{10}^{t-1})) \oplus z_{r-1}^{t-1}(1 \oplus A_7^{t-1} \oplus maj(A_3^{t-1}, A_7^{t-1}, A_{10}^{t-1})),$$
$$\text{for } r = 1 \cdots 22$$

and:

$$z_0^t = z_0^{t-1} \cdot (A_7^{t-1} \oplus maj(A_3^{t-1}, A_7^{t-1}, A_{10}^{t-1})) \oplus feedbackR_3^{t-1}(1 \oplus A_7^{t-1} \oplus maj(A_3^{t-1}, A_7^{t-1}, A_{10}^{t-1}))$$

where the feedback function for R2 is defined as it follows: $feedbackR_3^{t-1} = z_7^{t-1} \oplus z_{20}^{t-1} \oplus z_{21}^{t-1} \oplus z_{22}^{t-1}$

With these equations, it is possible to write the system of equations that defines the cipher A5/2, having as variables the internal states of the registers R1, R2 and R3, summing 64 variables. The internal state of R4 does not belong to the system variables because it will be guessed, as it takes only 216 possible values.

5. Existing attacks on A5/2 cipher

As we have seen, the security of A5/2system is based on the dificulty of solving large systems of multivariate quadratic equations.

5.1 Attacks based on solving quadratic equations system by linearization

In August 1999, a group of U.S. researchers claims to have broken the algorithm A5/2, using a single PC, within few seconds. This attack requires two frames of plain text but provided that these two will be at exactly 1326 frames apart. This rather unrealistic requirement makes the attack less feasible. Another attack is proposed in [7] that requires 4 frames of known plaintext. Although this attack does not recover the internal state of algorithm, yet is able to recover the rest of the conversation.

The most important attack on A5 algorithm is that proposed by a team of researchers from Israel: Elad Barkan, Eli Biham and Nathan Keller. They started with an attack on A5/2 which is the best time complexity, being the first ciphertext-only attack; then, they took the general idea and extended it to the A5/1 algorithm, also obtaining outstanding results. The attack is a passive one in which the attacker does not need to transmit any data and is based on an important observation related to the GSM system, namely the application of error correcting codes on the text before encryption. They have exploited this and built an attack that includes two phases: a preprocessing one which needs computing power and time very high (in a network of 1000 computers, preprocessing time reaches 8 months) and a real time stage that requires about 14 minute on a PC.

5.2 The system of equations for the Barkan case

All these attacks so far are based on the idea of solving the system of multivariate algebraic equation by linearization. The

best of these attacks requires four frames of plain text . In the case of the attack with four frames of plain text, the value of R4 is guessed and each output bit is written as a quadratic term 4 depending on R1, R2 and R3. The authors describe a method of writing each output bit as a quadratic term having as variables the states of R1, R2 and R3 from the first frame. With these bits coming from four frames, they built a system of quadratic equations solved by linearization, recovering the initial values for R1, R2 and R3. For each of the 216 possible values for R4, a system of equations is built and solved until it is found a consistent solution. After linearization, the system grows to 656 variables; taking into account that not all the variables show interest, as they are derived by linearization from products of variables, was found to be suficient over 450 independent linear equations that can be solved by Gauss elimination.

5.3 Briey about Groebner bases

For basic definitions, see [5]. Let K be a field and K[x_1, …x_n] the ring of multivariated polynomials. We are interested in solving systems of equations of the form:

$$\begin{cases} f_1(x_1,\ldots,x_n)=0 \\ \vdots \\ f_m(x_1,\ldots,x_n)=0 \end{cases}$$

where f_1, …, f_m are n-variate polynomials from the polynomial ring K[x_1, …, x_n].
Let I be the ideal generated by the polynomials f_1, …, f_m. A Groebner base is a particular kind of generating subset of an ideal I in a polynomial ring that behaves well towards certain orders defined on monomials. The main rules for ordering monomials in a polynomial are listed below, after being introduced a few concepts and terms that appear in the literature dedicated Groebner bases.

Observation 5.1. *The power product is the product of variables* $\{x_1^{\beta_1} * \cdots * x_n^{\beta_n} | \beta_i \in \mathbb{N}, i \leq i \leq n\}$. *The degree of the term of a polynomial is the sum of the term exponents from the power product. The term degree of a polynomial f is the sum of term exponents from the power product. The degree of a polynomial, denoted by deg(f) is the highest degree of all the terms of f. The leading term, denoted by lt(f) is the highest degree term. The leading coefficient of f is the coefficient of the leading term; the power product of the leading term is called the leading power product lpp(f).*

i Lex(lexicographic ordering): For two terms $t_1, t_2 \in \mathbb{T}^n$ is defined $t_1 \geq_{Lex} t_2$ if and only if the first non-zero component of $log(t_1) - log(t_2)$ is positive or $t_1 = t_2$; for instance,
$x_1^3 x_2^2 x_3^4 >_{Lex} x_1^3 x_2^2 x_3$ si $x_1 x_3 > x_2^2$.

ii DegLex(degree-lexicographic term ordering): For $t_1, t_2 \in \mathbb{T}^n$ we say that $t_1 \geq_{DegLex} t_2$ if $deg(t_1) > deg(t_1)$ or $deg(t_1) = deg(t_1)$ and $t_1 \geq_{Lex} t_2$; for instance, $x_1 x_2^2 x_3^3 >_{DegLex} x_1^2 x_2^2$.

iii DegRevLex (degree-reverse lexicographic term ordering): For $t_1, t_2 \in \mathbb{T}^n$ we say that $t_1 \geq_{DegRevLex} t_2$ if $deg(t_1) > deg(t_1)$ or if $deg(t_1) = deg(t_1)$ and the last non-zero component of $log(t_1) - log(t_2)$ is negative or if $t_1 = t_2$; for example, $x_1 x_5^2 x_2^3 >_{DegRevLex} x_1^4 x_2 x_3^3$.

Definition 5.1. *A finite set of nonzero elements of I G is called a Groebner basis for (f_1, …, f_m) if for any f ∈ I there exists g ∈ G so that lt(g) divides lt (f). The next proposition tells us how to use Groebner bases in order to solve a system of polynomial equations over Z_2:*

Proposition 5.1. *The Groebner basis of the ideal (f_1, …, f_m, x^2_1- x_1, …, x^2_n-x_n) in the ring Z/2[x_1, …, x_n] describes all the solutions of the system of polynomial equations in that ring. Particular cases:*
1. The system has no solution if and only if G = [1] (where G is the Grobner basis)
2. The system has exactly one solution if and only if G = [x_1 - a_1, …, x_n - a_n] where a_i ∈ (Z)/2. Then (a_1, …, a_n) is the solution of the system in the ring (Z)/2.

This proposition tells us we need to add another equation of type $x^2_i = x_i$ for each variable. Therefore, we have to calculate a Groebner basis for m + n polynomials and n variables. This can be only favorable

ISSN 2067 – 4074

considering that the more a system is over determined the easier is to solve.
The above result greatly simplifies the computation and the obtaining of the result by the fact that it is not necessary to apply a whole existing algorithm in literature for finding solutions of the system; these are described in a very natural manner by the reduced Grobner basis calculated.

6. Experimental results

We recall that the A5 algorithm returns the 114-bit encryption key that is used for encryption by a XOR operation. The attack that we discuss in this paper, being a known plaintext attack, implies that for some frames of ciphertext we know the corresponding plain text, which allows finding the keys used for encryption. With this, all our effort is headed for finding the session key K_c.
In order to simplify the attack so it can be run on an ordinary personal computer (3GHz CPU speed and memory 4 GB RAM), for known plaintext attack I used a single frame which means that will result 114 equations and it is assumed to be known the value of the register R4 after key setup phase and before the 99 cycles. Normally, the value of R4 is guessed, that is, are tested all the 216 possible values, repeating the above procedure until it is found a consistent solution. In this paper, we made only one of the 216 tests, being interested, particularly, in the computation time of the Groebner basis which gives the solution of the polynomial equation system.
The system will have 61 variables, the initial values (immediately after the key setup) of the first 3 registers. With formulas for the bits of each register at some point and also the formula for an output bit at the appropriate time, the 114 equations are obtained simply: as the right member will be 0 and in the left member will be the difference between the output bit given by the formula and the value of bit from the encryption key that can be recovered by a simple XOR operation. Then, the 114 expressions obtained together with the 61 expressions of the form $x_i - x^2_i$ are considered the generators of the ideal which has the reduced Groebner basis calculated with respect to the ordering of DegRevLex, and then it is considered the Grobner basis corresponding to the lexicographic order which describes the system solution.
For implementation, I initially tested several algebraic software (available free on the Internet) that provide the foundation to work with Grobner basis: Cocoa, Maple working with GB and FGb,and Sage. Finally, I limited the tests to CoCoA [9] and Sage [10] (the two software's that best responded to the challenge made), being comparatively used to test the speed of response achieved. [1] have successfully used the MAGMA software which contains a very eficient implementation of the algorithm F4. Unfortunately, it is not freely available on the Internet but I still managed to provide in this paper some results that I obtained by testing Magma.
Knowing that working with Grobner bases is not easy in terms of complexity, the time required to calculate the Grobner basis for the current system was not a surprise. Thus, in order to solve the system of initial equations 114 (to which were added other 61 equations) with 61 variables, I proceeded gradually:
i) I considered initially, from the 3 registries unknown (whose bits we want to find out), first register unknown and the other 2 registers known (recall that the 4th register we consider known throughout the attack), therefore the system turned into one with only 116 + 61 equations and 17 variables; the time necessary to solve the reduced system in Cocoa is 8 seconds (Sage requires about the same time).
ii) The next step was to consider known another unknown register (the second one), which expanded the number of variables to 38; in this situation, the difference between the algorithms used for calculating Groebner bases is huge: Coco and Sage (the latter offers a better result, in time) requires more than one hour (1h 30 min to Sage), while Magma offers by far the best performance: 37 seconds;
iii) The last step, the most important, includes even the problem to be solved:

the system of 61 variables. Here CoCoA and Sage fail after several hours of calculation, while Magma finds the solution in 5 days running on 60 processors in the cluster.
[1] state that the equations attached to any three frames in Magma can be calculated in only 2-3 seconds. If this is true, then it's an amazing result and a great saving of time: so the more ecuations the system has, the easier is to find its solution.

7. Conclusions

A plain text attack on A5/2 algorithm, that uses the technique of Groebner basis to solve the system of multivariate polynomial equations generated by the cipher, needs 3 frames of plain text. Magma, an algebraic software that works with Groebner bases, is said [1] to be able to calculate the solution of the 6 system of 114 x 3 polynomial equations with 61 variables in only 2-3 seconds; The experimental results I made concerning this problem conclude the following: a system with 116 + 61 equations and 17 variables needs 8 seconds in CoCoA and is done almost instantly in Magma; the same system with 38 variables needs 1h 30 min in CoCoA and only 37 seconds in Magma; for the complete system with 61 variables (the real one we need to solve) Magma finds the solution in 5 days running on 60 processors in the cluster; the other software just crashes.

References

[1] Mehreen Afzal, Ashraf Masood, Naveed Shehzad Improved Results on Algebraic Cryptanalysis of A5/2, Global E-Security. Springer Berlin Heidelberg, 2008

[2] Elad Barkan, Eli Biham, Nathan Keller Instant Ciphertext-Only Cryptanalysis of GSM Encrypted Communication, (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 600616. Springer, Heidelberg (2003)

[3] Nicolas Courtois, Alexander Klimov, Jacques Patarin, Adi Shamir Eficient Algorithms for Solving Overdefined Systems of Multivariate Polynomial Equations, EUROCrYPT, 2000

[4] Slobodan Petrovic, Amparo Fuster-Sabater : Cryptanalysis of the A5/2 Algorithm, IACR ePrint Report 200/52 (2000)

[5] Martin Kreuzer, Lorenzo Robbiano Computational Commutative Algebra 2, Springer-Verlag Berlin Heidelberg, 2005

[6] Marc Briceno, Ian Goldberg, David Wagner A Pedagogical Implementation of the GSM A5/1 and A5/2 Voice Privacy Encryption Algorithms (1999),

[7] Slobodan Petrovic, Amparo Fuster-Sabater CRYPTANALYSIS OF THE A5/2 ALGORITHM

[8] Mehreen Afzal, Ashraf Masood On Generating Algebraic Equations for A5-Type Key Stream Generator, Trends in Intelligent Systems and Computer Engineering Series. LNEE, vol. 6, pp. 443451. Springer, US; An extended version of Algebraic Attack on A5-Type Irregularly Clocked Key Stream Generator. In: Proc. International Multiconference of Engineers and Computer Scientists-IMECS 2007, IAENG (March 2007)

[9] http://cocoa.dima.unige.it/

[10] http://www.sagemath.org/

[11] MAGMA Computational Algebra System, http://magma.maths.usyd.edu.au/