Departamento de Matemáticas, Facultad de Ciencias
Universidad Autónoma de Madrid

Aspectos teóricos y computacionales del problema inverso de Galois

Trabajo de fin de grado

Grado en Matemáticas

Autor: Tomás Senovilla Polo
Tutor: Enrique González Jiménez

Curso 2018-2019

Resumen

Dados un cuerpo K y un grupo finito G, el problema inverso de Galois consiste en determinar si existe una extensión de Galois L/K tal que posea a G como grupo de Galois. A lo largo de este proyecto, vamos considerar $K = \mathbb{Q}$ y ciertas familias de grupos tratando de dar una respuesta afirmativa al problema. Para cada familia de grupos vamos a estructurar el estudio en dos partes bien diferenciadas: primero, abordaremos las demostraciones puramente matemáticas para asegurarnos que es posible resolver el problema en cada caso; después, mediante el uso de la informática ofreceremos una forma eficiente de resolver nuestro problema.

Abstract

Given a field K and a group G, the inverse Galois problem consist in find a Galois extension L/K such that G is the Galois group of L. Throughtout this project we are going to let K be $\mathbb{Q}$ and G some group families. We are going to deal with the problem, for each familiy, in two different ways: firstly, by carrying out the pure mathematical proofs of the statements, in order to ensure that the problem may be solved; then, we are going to present an effective way to solve the problem by the use of a computer.

Índice general

CAPÍTULO 1
Introducción y preliminares

1.1. Introducción

El problema inverso de Galois aparece de forma natural dentro de la Teoría de Galois. Recordemos que el Teorema Fundamental de la Teoría de Galois ofrecía una correspondencia biyectiva entre las subextensiones de una extensión algebraica K/E y los subgrupos del grupo de Galois de K sobre E. Una vez establecida esta correpondencia tratabamos de encontrar el grupo de Galois de una extensión K/E dada. El problema inverso de Galois consiste en determinar, dados un grupo finito G y un cuerpo E, si existe una extensión de Galois K/E tal que $Gal(K/E) \cong G$.

A pesar de que el problema no está resuelto en general, sí que podemos dar una respuesta en ciertos casos, dependiendo del cuerpo base sobre el que trabajemos.

Si consideramos el cuerpo $\mathbb{C}(t)$, donde t es una indeterminada, el problema inverso de Galois tiene respuesta afirmativa, aunque este resultado no será explicado en el presente texto, ya que la prueba excede con creces el nivel del mismo.

Por contra, si trabajamos sobre un cuerpo finito la respuesta es negativa, ya que es sabido que el grupo de Galois de cualquier extensión finita de un cuerpo finito es cíclico. Este resultado no es difícil y el lector interesado puede encontrar un esquema de la demostración en [4, Sección 46.4].

A partir de ahora, y salvo que se especifique lo contrario, vamos a tratar de dar respuesta al problema inverso de Galois sobre el cuerpo de los números racionales. En este caso, estamos tratando con un problema abierto, y nosotros nos vamos a centrar en dar una respuesta afirmativa para ciertas familias de grupos. En los grupos tratados vamos a proporcionar una respuesta total al problema, demostrando primero la posibilidad de resolución, para pasar después a construir métodos eficaces que resuelvan el problema.

Antes de eso, en la siguiente sección, vamos a dedicarnos a repasar algunas cuestiones fundamentales acerca de la Teoría de grupos y anillos, así como de la Teoría de Galois. Suponemos que el lector posee ciertos conocimientos dentro de esta materia, por lo que no se incluiran muchas demostraciones, que pueden ser encontradas en cualquier libro de iniciación a la Teoría de Galois. Otras se ofrecerán en el Apéndice A.

1.2. Resultados preliminares

Uno de los resultados más importantes de la teoría de grupos abelianos finitos es el relacionado con la estructura de estos. Usaremos este resultado especialmente en el capítulo relacionado con resolver el problema inverso de Galois para grupos abelianos finitos.

Teorema 1.2.1. *Sea G un grupo abeliano finito. Entonces G es isomorfo a un producto directo de grupos cíclicos. En particular, $G \cong \mathbb{Z}/n_1\mathbb{Z} \times \cdots \mathbb{Z}/n_s\mathbb{Z}$ con $n_i | n_{i+1}$, $\forall i < s$.*

El siguiente resultado trata sobre polinomios definidos en un cuerpo finito y será útil en capítulos posteriores. Su demostración puede ser encontrada en el Apéndice A.

Proposición 1.2.2. *Sea $f(x) \in \mathbb{F}_p[x]$ un polinomio de grado n tal que $p \nmid na_n$ donde a_n es el coeficiente del término de mayor grado de $f(x)$. Supongamos que $f(x)$ tiene una raíz repetida en $\mathbb{F}_p[x]$, entonces $mcd(f, f') \neq 1$, donde $f'(x)$ representa la derivada del polinomio $f(x)$.*

Lo verdaderamente interesante de este resultado es su contrarrecíproco, esto es, si estamos trabajando con un polinomio en $\mathbb{F}_p[x]$ bajo las condiciones del enunciado y somos capaces de demostrar que tanto dicho polinomio como su derivado son coprimos, entonces tendremos que todas las raíces del polinomio en $\mathbb{F}_p[x]$ son simples.

Vamos ahora a enunciar algunos resultados de Teoría de Galois que necesitaremos durante nuestro estudio del problema inverso. Comenzamos definiendo los automorfismos fundamentales entre dos extensiones algebraicas de un cuerpo K. Estos automorfismos serán importantísimos, ya que como sabemos por la Teoría de Galois, cuando trabajamos con extensiones algebraicas construimos el grupo de Galois a través de estos isomorfismos.

Teorema 1.2.3. *Sea F un cuerpo y α, β algebraicos sobre F. La transformación $\psi_{\alpha,\beta} : F(\alpha) \to F(\beta)$ definida de tal modo que todo elemento de F queda fijo y $\psi_{\alpha,\beta}(\alpha) = \beta$ es un isomorfismo sí y solo sí α y β son conjugados sobre F, es decir, si $Irr_F(\alpha) = Irr_F(\beta)$.*

Observamos que estos isomorfismos están definidos para cualquier elemento de $F(\alpha)$ ya que la base de la extensión como espacio vectorial es totalmente dependiente de α.

Definición 1.2.4. *Decimos que un automorfismo φ de un cuerpo E deja fijo un subcuerpo $F \subseteq E$ si $\forall a \in F$ se tiene que $\varphi(a) = a$.*

Se tiene que el conjunto de todos los automorfismos de un cuerpo E es un grupo con la composición de funciones y que si E/F es una extensión de cuerpos, el conjunto de los automorfismos de E que dejan fijo a F es un subgrupo de dicho grupo. Este subgrupo es lo que a partir de este momento denotaremos por $Gal(E/F)$ y diremos que es el grupo de Galois de E sobre F.

Definición 1.2.5. *Sea F un cuerpo y S una colección finita de polinomios en $F[x]$. Decimos que una extensión E/F es el cuerpo de descomposición de S sobre F si es la menor extensión de F que contiene a todos los ceros de cada elemento de S. Una extensión E/F es un cuerpo de descomposición sobre F si es el cuerpo de descomposición de algún conjunto de polinomios en $F[x]$.*

Definición 1.2.6. *Sea F un cuerpo y $f(x) \in F[x]$. Decimos que:*

- *$f(x)$ es separable sí y solo sí todos sus ceros tienen multiplicidad 1.*
- *$F(\alpha)$ es una extensión separable sí y solo sí $Irr_F(\alpha)$, su polinomio irreducible sobre el cuerpo F, es separable.*
- *Un elemento α algebraico sobre F es separable sí y solo sí $F(\alpha)$ es separable.*
- *Una extensión finita E/F es separable sobre F sí y solo si todo elemento $\alpha \in E$ es separable sobre F.*
- *Una extensión E/F es de Galois sí es un cuerpo de descomposición sobre F y además es separable.*

Para ver que una extensión es un cuerpo de descomposición debemos encontrar un conjunto de polinomios del cuerpo base que se descomponga en nuestra extensión, lo cual no tiene porque ser necesariamente fácil. Comprobar que una extensión es separable puede ser también complicado, afortunadamente, como ya hemos dicho, nos interesan extensiones finitas de $\mathbb{Q}$ y podemos afirmar que estas son siempre separables gracias al siguiente resultado, cuya demostración se puede encontrar en [4, Teoremas 43.4 y 43.5].

Teorema 1.2.7. *Todo cuerpo de característica 0 o finito es perfecto, esto es, toda extensión finita de dicho cuerpo es separable.*

Enunciamos finalmente el Teorema Fundamental de la Teoría de Galois. La demostración no se incluye aquí, no obstante el lector interesado puede encontrarla en cualquier libro de iniciación a la Teoría de Galois. En esta enunciación del teorema se ha seguido principalmente el orden propuesto en [4].

Teorema 1.2.8. *(Teorema Fundamental de la Teoría de Galois) Sea K una extensión de Galois de un cuerpo F y sea el grupo de Galois de dicha extensión $Gal(K/F)$. Sea E un subcuerpo intermedio, esto es, $F \subseteq E \subseteq K$ y $Gal(K/E)$ el subgrupo del grupo de Galois que deja fijo E. Entonces existe una correspondencia biyectiva entre las subextensiones de K y los subgrupos del grupo de Galois tal que:*

- *La imagen de E es $Gal(K/E)$. Reciprocamente, el subgrupo del grupo de Galois que deja fijo E tiene imagen E.*
- *$[K : E] = |Gal(K/E)|$, es decir, el grado de la extensión es igual que el orden del grupo de Galois correspondiente.*

- *E es de Galois sí y solo sí $Gal(K/E)$ es un subgrupo normal del grupo de Galois. En este caso se tiene que*

$$Gal(E/F) \cong Gal(K/F)/Gal(K/E)$$

CAPÍTULO 2

El problema inverso de Galois para grupos abelianos finitos

Cuando se afronta el problema inverso de Galois por primera vez, lo más natural es empezar por considerar los grupos abelianos finitos, debido a que su estructura es conocida en general. La solución al problema en este caso es sencilla y se basa en dos conceptos: la estructura de las extensiones ciclotómicas y el Teorema de Dirichlet acerca de los números primos en progresiones aritméticas.

Recordemos antes de empezar, de manera breve, el concepto de extensión ciclotómica. Sea ϵ una raíz primitiva n-ésima de la unidad, definimos el n-ésimo polinomio ciclotómico como $\Phi_n(x) = \prod_{i=1}^{\phi(n)} (x - \epsilon_i)$, donde ϵ_i son las $\phi(n)$ raíces primitivas n-ésimas de la unidad. Aquí $\phi(n)$ hace referencia a la función ϕ de Euler. Se tiene que $\Phi_n(x)$ es un polinomio mónico e irreducible en $\mathbb{Q}[x]$. Además, la construcción del polinomio ciclotómico nos da de manera fácil la identidad $x^n - 1 = \prod_{d|n} \Phi_d(x)$.

La $n-$ésima extensión ciclotómica se define como el cuerpo de descomposición del $n-$ésimo polinomio ciclotómico, y debido a que las raíces de este polinomio son todas potencias de ϵ, para ϵ una raíz primitiva $n-$ésima, la $n-$ésima extensión ciclotómica es una extensión simple de $\mathbb{Q}$ definida como $\mathbb{Q}(\epsilon)$.

2.1. Existencia de una extensión de Galois con grupo de Galois un grupo abeliano finito

En esta sección vamos a demostrar que dado un grupo abeliano finito G existe una extensión de Galois $K/\mathbb{Q}$ cuyo grupo de Galois es G. La demostración es totalmente constructiva y se basa en extensiónes ciclotómicas. Adelantándonos a lo que vendrá después queremos hacer notar que esta construcción no es única y de hecho existen infinitas formas de construir una extensión con el grupo de Galois que buscamos.

Un ejemplo de esta no unicidad es trivial y consiste en que tanto $\mathbb{Q}(\sqrt{2})$ como $\mathbb{Q}(\epsilon)$ donde ϵ es una raíz primitiva 3-ésima de la unidad, poseen el mismo grupo de Galois, a saber $\mathbb{Z}/2\mathbb{Z}$. La extensión $\mathbb{Q}(\sqrt{2})$ no es ciclotómica, lo que muestra que la construcción que vamos a hacer no es única, sin embargo, es quizás la más sencilla.

Antes de empezar, presentamos dos resultados que serán útiles a lo largo de la sección. La demostración de ambos puede ser encontrada en el apendice A.

Lema 2.1.1. *Sea G un grupo abeliano y sean $x_1, x_2, ...x_m \in G$ tales que $o(x_i) = n_i$ y $mcd(n_i, n_j) = 1$ si $i \neq j$. Se tiene que $o(x_1 \cdots x_m) = \prod_{i=1}^{m} n_i$.*

Lema 2.1.2. *Sea $(F, +, \cdot)$ un cuerpo y sea $(F^*, \cdot)$ su grupo multiplicativo. Sea $G \leq F^*$ finito, entonces G es cíclico.*

También será importante el siguiente resultado de Teoría de Galois, que muestra como es el grupo de Galois de la n-ésima extensión ciclotómica.

Proposición 2.1.3. *Sea ϵ una raíz primitiva n-ésima de la unidad, se verifica que $Gal(\mathbb{Q}(\epsilon)/\mathbb{Q}) \cong (\mathbb{Z}/n\mathbb{Z})^*$*

Demostración. Sabemos por Teoría de Galois que $Gal(\mathbb{Q}(\epsilon)/\mathbb{Q})$ está formado por los automorfismos de $\mathbb{Q}(\epsilon)$ y que estos automorfismos quedan totalmente determinados por la imagen de ϵ. Sabemos además que estas imagenes deben ser precisamente las demás raíces primitivas n-ésimas de la unidad[1]. Calcular estas raíces es muy sencillo a partir de ϵ, pues δ es raíz primitiva $n-$ésima de la unidad sí y solamente si $\delta = \epsilon^j$, donde $mcd(j, n) = 1$.

Por tanto, hemos visto que $Gal(\mathbb{Q}(\epsilon)/\mathbb{Q}) = \{\sigma_i : \sigma_i(\epsilon) = \epsilon^i,\ mcd(i, n) = 1\}$.

Sea ahora $f : Gal(\mathbb{Q}(\epsilon)/\mathbb{Q}) \to (\mathbb{Z}/n\mathbb{Z})^*$ tal que $f(\sigma_i) = i$. f es claramente biyectivo y como $\sigma_i\sigma_j = \sigma_{ij}$, se tiene que $f(\sigma_i\sigma_j) = ij = f(\sigma_i)f(\sigma_j)$. Por tanto f es un isomorfismo de grupos.

□

Corolario 2.1.4. *$| Gal(\mathbb{Q}(\epsilon)/\mathbb{Q}) | = \phi(n)$, donde $\phi(n)$ es la función ϕ de Euler.*

El siguiente teorema es la herramienta clave para alcanzar nuestro objetivo, por ello la demostración se realiza con todo lujo de detalles. Es un caso particular del Teorema de Dirichlet de los primos en progresiones aritméticas.

Teorema 2.1.5. *Para cada $n \in \mathbb{N}$, existen infinitos primos p tales que $p \equiv 1$ (mód n).*

Demostración. La demostración consiste en dos pasos independientes que convergen finalmente al resultado.

[1] Esto es fácil de ver, pues si σ es uno de los automorfismos que buscamos se tiene que $1 = \sigma(1) = \sigma(\epsilon^n) = (\sigma(\epsilon))^n$ y además $(\sigma(\epsilon))^i \neq 1\ \forall i < n$, pues si no se tendría que $\epsilon^i = 1$, contradiciendo que esta raíz es primitiva.

Veamos primero que si $g(x) \in \mathbb{Z}[x]$ es un polinomio se tiene que hay infinitos primos que dividen a algún miembro del conjunto $\{g(0), g(1), ...\}$. Supongamos primero que el término independiente de $g(x)$ es 1. Si tuvieramos que el número de primos que dividen a algún miembro de $\{g(0), g(1), ...\}$ es finito, digamos estos primos son p_1,p_2,...,p_r, considerando el polinomio $g(p_1p_2...p_rx)$ tenemos que debe de existir algún entero positivo x_0 tal que $\exists p \notin \{p_1,p_2,...,p_r\}$ primo con $p \mid g(p_1p_2...p_rx_0)$. Si el término independiente de $g(x)$ es $a \neq 1$, los coeficientes del polinomio $g(ax)$ son divisibles por a y por tanto $g(x) = a \cdot h(x)$ donde $h(x)$ tiene término independiente igual a 1. Como sabemos que los primos que dividen a los términos de la sucesión $\{h(0), h(1), ...\}$ son infinitos, también lo son los que dividen a los $g(a \cdot x)$ y por tanto los que dividen a algún miembro de $\{g(0), g(1), ...\}$ son infinitos, pues los $g(a \cdot x)$ están en esa lista.

Sea ahora $n \in \mathbb{N}$ y p un primo que no divide a n. Vamos a ver que $p \mid \Phi_n(c)$ para algún entero c sí y solamente si el orden de c en $\mathbb{F}_p$ es n. Para ver esto, consideramos que $p \mid \Phi_n(c)$, entonces se tiene que $p \mid c^n - 1$ y por tanto $c^n \equiv 1 \pmod{p}$. Si el orden de c en $\mathbb{F}_p$ fuera $d < n$, en particular $d \mid n$ y como $c^d \equiv 1 \pmod{p}$ se tendría, ya que $x^d - 1 = \prod_{e \mid d} \Phi_e(x)$, que c es una raíz de $\Phi_n(x)$ y de $\Phi_e(x)$ en $\mathbb{F}_p[x]$ para algún $e \mid n$, $e < n$, es decir, se tiene que $x^n - 1 = \prod_{d \mid n} \Phi_d(x)$ tiene una raíz múltiple en $\mathbb{F}_p$, pero esto es imposible pues $x^n - 1$ y su derivada son primos relativos en $\mathbb{F}_p[x]$ y como p no divide a n, tenemos que $x^n - 1$ tiene todas sus raíces distintas en, lo que se sigue de la Proposición 1.2.2. Por tanto el orden de c es n. Recíprocamente, si $o(c) = n$, se tiene que $c^n - 1 \equiv 0 \pmod{p}$. Además dado $d < n$ se tiene $c^d \not\equiv 1 \pmod{p}$. Por tanto c no es raíz de $\Phi_d(x)$ en $\mathbb{F}_p[x]$ para estos d y debe ser raíz de $\Phi_n(x)$ en $\mathbb{F}_p[x]$.

Finalmente, hay infintos primos dividiendo a $\Phi_n(c)$ para $c \in \mathbb{N}$. Evidentemente hay infinitos de esos primos p que no dividen a n y por tanto infinitos primos p tales que c con orden n en $\mathbb{F}_p$. Esto es equivalente[2] a que existen infinitos primos p con $p \equiv 1 \pmod{n}$. □

A partir de este resultado ya podemos hallar la solución a nuestro problema en grupos cíclicos finitos. Recordemos que cualquier grupo abeliano finito es producto directo de grupos ciclícos finitos, por lo que este es un punto de partida razonable.

Teorema 2.1.6. *Sea $G = \mathbb{Z}/n\mathbb{Z}$. Existe una extensión $K/\mathbb{Q}$ tal que K es de Galois y $Gal(K/\mathbb{Q}) \cong G$.*

Demostración. Por el Teorema 2.1.5, sabemos que $\exists p$ primo tal que $p - 1 = n \cdot m$ para algún $m \in \mathbb{N}$. Sea ϵ una raíz p-ésima de la unidad[3], sabemos por la Proposición 2.1.3 que $J = Gal(\mathbb{Q}(\epsilon)/\mathbb{Q}) \cong \mathbb{F}_p^*$, que es cíclico debido al Lema 2.1.2 y que $\mathbb{F}_p$ es un cuerpo. Como es cíclico, sabemos por Teoría de Grupos que existe un único $H \leq J$ tal que $|H| = m$, llamemos K al cuerpo fijo por H. Como J es abeliano, H es normal y por el Teorema Fundamental de la Teoría de Galois se tiene que K es una extensión

[2] Si p es tal que $c^n \equiv 1 \pmod{p}$, entonces $n \mid (p-1)$

[3] En particular esta raíz es primitiva por ser p primo.

de Galois sobre $\mathbb{Q}$. Además, de nuevo por el Teorema Fundamental y el hecho de que J es un grupo cíclico[4] se tiene que $Gal(K/\mathbb{Q}) \cong J/H \cong G$. □

Para visualizar bien como funciona esta construcción vamos a realizar un ejemplo sencillo, hallando dos extensiones de Galois que tengan como grupo a $\mathbb{Z}/2\mathbb{Z}$. No se sorprenderá el lector de que una de las extensiones resultante sea precisamente la que mencionábamos al principio de la sección. El interés de construir dos distintas utilizando este método es mostrar como se pueden buscar tantas como queramos, aunque el coste de trabajo crece de manera considerable rápidamente si buscamos varias distintas.

Note el lector también que la segunda extensión generada es precisamente $\mathbb{Q}(\sqrt{5})$, lo que sabemos por la teoría de las extensiones ciclotómicas. Sin embargo, extensiones cuadráticas como $\mathbb{Q}(\sqrt{6})$ no se pueden construir utilizando este método. Esto se reduce al caso $\mathbb{Z}/2\mathbb{Z}$ y el hecho de que las extensiones ciclotómicas p-ésimas tienen una única subextensión de grado 2 sobre $\mathbb{Q}$ que es de la forma $\mathbb{Q}\left(\sqrt{(-1)^{\frac{p-1}{2}}p}\right)$.

Ejemplo 2.1.7. *Sea $G = \mathbb{Z}/2\mathbb{Z}$. Vamos a construir una extensión de Galois que tenga a G como grupo de automorfismos. Como en la demostración del Teorema 2.1.6, buscamos $p \equiv 1 \pmod 2$. Tomamos por ejemplo $p = 3$. Si ε es una raíz primitiva 3-ésima de la unidad, la extensión $\mathbb{Q}(\varepsilon)/\mathbb{Q}$ es de Galois y su grupo de automorfismos es precisamente G. Nótese que en este caso, el grupo H con el que cocientábamos en la demostración es el trivial (pues en este caso $3-1 = 2\cdot 1$). Sin embargo, observamos que $p = 5$ también es un primo congruente con $1 \pmod 2$. Si tomamos δ una raíz primitiva 5-ésima de la unidad, la extensión $\mathbb{Q}(\delta)/\mathbb{Q}$ es de Galois, y sabemos por el Lema 2.1.2 que su grupo de Galois es $\mathbb{Z}/4\mathbb{Z}$. Siguiendo la construcción de la demostración, existe $H \leq \mathbb{Z}/4\mathbb{Z}$ con orden 2, tal que si K es su cuerpo fijo, $K/\mathbb{Q}$ es de Galois y su grupo de Galois es G. En este caso, el grupo H tiene orden 2 pues $5-1 = 2\cdot 2$. Podemos observar que $K \neq \mathbb{Q}(\varepsilon)$, ya que por el comentario precedente al ejemplo, $\mathbb{Q}(\varepsilon) \cong \mathbb{Q}(\sqrt{-3})$, mientras que $K \cong \mathbb{Q}(\sqrt{5})$. Lo interesante de este ejemplo es que a pesar de haber escogido el grupo cíclico no trivial más sencillo, la extensión buscada no es única, y esto se debe precisamente a que la elección de p no es única.*

A la vista del ejemplo vemos que uno de los pasos más importantes es, siguiendo la notación del Teorema 2.1.6, recordar que m hemos utilizado para hallar el número primo. Más adelante veremos que recordar esos números es también fundamental cuando hablemos de un grupo abeliano finito en general.

Presentamos finalmente un último resultado para a continuación demostrar nuestro objetivo fundamental, el problema inverso de Galois para grupos abelianos finitos.

Lema 2.1.8. *Sean n_1,n_2,...$n_t \in \mathbb{N}$ primos entre sí. Sea ϵ_i una raíz primitiva n_i-ésima de la unidad. Entonces $\epsilon = \epsilon_1\epsilon_2...\epsilon_t$ es una raíz $n_1n_2...n_t$-ésima de la unidad y $Gal(\mathbb{Q}(\epsilon)/\mathbb{Q}) \cong (\mathbb{Z}/n_1\mathbb{Z})^* \times (\mathbb{Z}/n_2\mathbb{Z})^* \times \cdots \times (\mathbb{Z}/n_t\mathbb{Z})^*$*

Demostración. La demostración es casi inmediata gracias al trabajo que tenemos realizado. Como $(\mathbb{C}^*, \cdot)$ es abeliano, se sigue del Lema 2.1.1 que ϵ tiene orden $n_1n_2...n_t$,

[4] Lo que implica que J/H también lo es.

es decir, ϵ es una raíz primitiva $n_1n_2...n_t$-ésima de la unidad. Por la Proposición 2.1.3 obtenemos que

$$Gal(\mathbb{Q}(\epsilon)/\mathbb{Q}) \cong (\mathbb{Z}/n_1n_2\dots n_t\mathbb{Z})^* \cong (\mathbb{Z}/n_1\mathbb{Z})^* \times (\mathbb{Z}/n_2\mathbb{Z})^* \times \cdots \times (\mathbb{Z}/n_t\mathbb{Z})^*,$$

debido a que $mcd(n_i, n_j) = 1$ si $i \neq j$. □

Teorema 2.1.9. *(Solución al problema Inverso de Galois para grupos abelianos finitos) Sea G un grupo abeliano finito, entonces podemos construir una extensión de Galois $K/\mathbb{Q}$ tal que $Gal(K/\mathbb{Q}) \cong G$.*

Demostración. Como G es un grupo abeliano finito, podemos escribirlo como $\mathbb{Z}/n_1\mathbb{Z}\times \mathbb{Z}/n_2\mathbb{Z} \times \cdots \times \mathbb{Z}/n_t\mathbb{Z}$. Del Teorema 2.1.5 y su demostración sabemos que existen p_1,p_2,..p_t tales que existen únicos subgrupos $H_i \leq (\mathbb{Z}/p_i\mathbb{Z})^*$ con orden $m_i = \frac{p_i-1}{n_i}$ y se verifica que $(\mathbb{Z}/p_i\mathbb{Z})^*/H_i \cong \mathbb{Z}/n_i\mathbb{Z}$. Ahora, sea $\epsilon = \epsilon_1\epsilon_2...\epsilon_t$ como en el Lema 2.1.8 una raíz $p_1p_2\cdots p_t$-ésima de la unidad, sabemos por dicha proposición que $Gal(\mathbb{Q}(\epsilon)/\mathbb{Q}) \cong (\mathbb{Z}/p_1\mathbb{Z})^* \times (\mathbb{Z}/p_2\mathbb{Z})^* \times \cdots \times (\mathbb{Z}/p_t\mathbb{Z})^*$. Sea $H \leq Gal(\mathbb{Q}(\epsilon)/\mathbb{Q})$ el subgrupo correspondiente a $H_1 \times H_2 \times ... \times H_t$ y sea K su cuerpo fijo. Como $Gal(\mathbb{Q}(\epsilon)/\mathbb{Q})$ es abeliano, se tiene que H es normal y por el Teorema Fundamental de la Teoría de Galois, K es una extensión de Galois sobre $\mathbb{Q}$ y también que $Gal(K/\mathbb{Q}) \cong Gal(\mathbb{Q}(\epsilon)/\mathbb{Q})/H \cong G$, por como hemos construido H. □

Al igual que hicimos cuando demostramos el resultado correspondiente para grupos cíclicos, veamos un ejemplo para un grupo abeliano no cíclico. Aprovecharemos este ejemplo para motivar la siguiente sección.

Ejemplo 2.1.10. *Vamos a usar la construcción de la demostración para buscar una extensión de $\mathbb{Q}$ que sea de Galois y cuyo grupo de automorfismos sea $\mathbb{Z}/3\mathbb{Z} \times \mathbb{Z}/9\mathbb{Z}$. Como $7 \equiv 1 \pmod 3$ y $19 \equiv 1 \pmod 9$, tomamos una raíz primitiva 133-ésima de la unidad[5] ε y tenemos $\mathbb{Q}(\varepsilon)/\mathbb{Q}$ es de Galois y su grupo de Galois es isomorfo a $(\mathbb{Z}/7\mathbb{Z})^* \times (\mathbb{Z}/19\mathbb{Z})^*$. Sabemos [6] que existen únicos subgrupos $H_1 \leq (\mathbb{Z}/7\mathbb{Z})^*$, $H_2 \leq (\mathbb{Z}/19\mathbb{Z})^*$ ambos con orden 2 y tales que $(\mathbb{Z}/7\mathbb{Z})^*/H_1 \cong \mathbb{Z}/3\mathbb{Z}$, $(\mathbb{Z}/19\mathbb{Z})^*/H_2 \cong \mathbb{Z}/9\mathbb{Z}$. Si tomamos K como el cuerpo fijo de $H_1 \times H_2$, se tiene que es una extensión de Galois y $Gal(K/\mathbb{Q}) \cong \mathbb{Z}/3\mathbb{Z} \times \mathbb{Z}/9\mathbb{Z}$, por la construcción que hemos hecho en el Teorema.*

A raíz de este ejemplo, son pertinentes las observaciones siguientes:

1. Hemos utilizado la construcción de la demostración para afirmar que el grupo de Galois de K es el que buscamos. Sin embargo ver el isomorfismo no es fácil.

2. Hemos encontrado una extensión $K/\mathbb{Q}$ de Galois, pero solo sabemos que es el cuerpo fijo de un subgrupo del grupo de Galois de $\mathbb{Q}(\varepsilon)/\mathbb{Q}$ donde ε es una raíz primitiva 133-ésima de la unidad. Conocemos el grado de la extensión K por el

[5] Note el lector que al igual que para grupos cíclicos, esta elección de primos no es única. Lo verdaderamente interesante es que 7 y 19 son los primos más pequeños que nos valen en nuestra búsqueda y que en este caso tenemos que trabajar con la 133-ésima extensión ciclotómica.

[6] Recordemos que a la hora de encontrar los subgrupos que hemos buscado es importantísimo recordar las m_i que utilizamos, siguiendo la notación del Teorema 2.1.9. En este caso $m_1 = m_2 = 2$.

Teorema Fundamental de la Teoría de Galois ($[K : \mathbb{Q}] = 27 = |\mathbb{Z}/3\mathbb{Z} \times \mathbb{Z}/9\mathbb{Z}|$). Pero encontrar una definición de K como extensión simple de $\mathbb{Q}$ es mucho más complicado.

3. Las dos observaciones anteriores dan muestras de las dificultades que tiene la construcción explícita de la extensión y el isomorfismo requeridos en el caso de un grupo relativamente pequeño y manejable, de orden 27, y en el cual tenemos que considerar raíces 133-ésimas de la unidad[7]. Si consideramos un grupo más grande, la dificultad crece muy rápidamente. Es decir, intentar encontrar la extensión K y el isomorfismo de forma explícita a mano solo es posible en los casos más básicos.

2.2. Cálculo de un polinomio con grupo de Galois un grupo abeliano finito

En la sección anterior hemos visto que el problema inverso de Galois tiene solución afirmativa en el caso de los grupos abelianos finitos. No obstante, hallar dicha extensión a mano no es algo sencillo. Vamos a utilizar nuestro conocimiento acerca de que dicha extensión existe para calcularla utilizando el ordenador, aunque haremos una construcción distinta a la de la demostración, en un intento de agilizar el funcionamiento del programa.

En esta sección vamos a utilizar el programa informático de libre distribución Sage [7], que el lector interesado puede encontrar en la página web de los desarrolladores, a saber `http://www.sagemath.org`. Ayudándonos de Sage vamos a diseñar un programa que nos permita calcular de forma rápida un polinomio con grupo de Galois un grupo abeliano finito a elección del usuario[8]. En esta sección se supone que el usuario tiene conocimientos previos de este lenguaje de programación.

Presentaremos un programa que resuelva el problema inverso de Galois para grupos abelianos finitos y lo usaremos en un ejemplo para comprobar que funciona. Para definir nuestro programa, necesitaremos varios programas auxiliares que no se incluyen en esta sección para evitar que el lector pierda de vista el objetivo marcado. No obstante, si el lector siente curiosidad por la definición y el funcionamiento de estos programas le invitamos a que consulte el Apéndice B.

Empezamos trabajando con grupos cíclicos finitos. En este caso, la programación será un calco total del algoritmo descrito en la sección anterior. Recordemos que un punto muy importante a la hora de construir la extensión cuyo grupo de Galois era $\mathbb{Z}/n\mathbb{Z}$, era recordar el m que buscábamos para que $p = n \cdot m + 1$ fuera primo. Por tanto lo primero que vamos a hacer es escribir un programa que busca el menor $m \in \mathbb{N}$ con esta propiedad a partir de una fijada.

[7] Como mínimo, pues hemos visto que la elección de los primos no es única. En este caso, 7 y 19 es la elección optima en cuanto a minimizar los grupos con los que trabajamos.

[8] Decimos que un polinomio $f \in K[x]$ tiene grupo de Galois G si $G = Gal(L/K)$, donde L es el cuerpo de descomposición de f. Lo denotamos por $Gal_K(f)$.

```
def generam(n,m):
   while (n*m+1).is_prime()==False:
      m = m+1;
   return m
```

Presentamos ahora un programa cuyo input son n, haciendo referencia al grupo $\mathbb{Z}/n\mathbb{Z}$, y s, haciendo referencia al punto desde el cual buscaremos una m tal que $n \cdot m + 1$ sea primo. Su explicación constituye el comienzo del Apéndice B.

```
def GalInvProblemCyclicGroup(n,s):
   x = polygen(QQ,'x');
   m = generam(n,s);
   p = (n*m)+1;
   if m==1:
      return cyclotomic_polynomial(p)
   Galp = CyclotomicField(p).galois_group();
   H=Galp;
   i=0;
   while H.order()!=m:
      H=Galp.subgroup([Galp[i]]);
      i=i+1;
   K=H.fixed_field()[0];
   return [K.defining_polynomial(),m]
```

Nuestro objetivo es escribir un programa que tenga como input una lista $[n_1, ..., n_l]$ y como salida un polinomio irreducible sobre $\mathbb{Q}$ cuyo grupo de Galois sea $\mathbb{Z}/n_1\mathbb{Z} \times \cdots \times \mathbb{Z}/n_l\mathbb{Z}$. Un primer intento de lograrlo sería, al igual que con los grupos cíclicos, el de repetir el algoritmo usado para demostrar la existencia. Sin embargo, este método no es muy eficiente implementado en Sage. Alternativamente, vamos a aprovecharnos de que sabemos que existen extensiones que verifican nuestro propósito para construir una de ellas basándonos en el siguiente resultado de Teoría de Galois:

Proposición 2.2.1. *Sean f_1, f_2 polinomios irreducibles en $\mathbb{Q}[x]$ cuyos grupos de Galois son $\mathbb{Z}/n_1\mathbb{Z}, \mathbb{Z}/n_2\mathbb{Z}$ respectivamente. Sea N_1 el cuerpo de descomposición de f_1 y supongamos que f_2 es irreducible en $N_1[x]$. Si N_2 es el cuerpo de descomposición de $\{f_1, f_2\}$, entonces su grupo de Galois es $\mathbb{Z}/n_1\mathbb{Z} \times \mathbb{Z}/n_2\mathbb{Z}$.*

Demostración. Sabemos que el grupo de Galois de N_1 es $\mathbb{Z}/n_1\mathbb{Z}$. Sea M el cuerpo de descomposición de f_2 sobre $\mathbb{Q}$, su grupo de Galois es por tanto $\mathbb{Z}/n_2\mathbb{Z}$. Podemos hallar el grupo de Galois de N_2 de manera sencilla componiendo cada automorfismo de ambos grupos y esta composición está bien definida en N_2 pues es una extensión al mismo tiempo de N_1 y M. Por la manera en la que hemos obtenido el grupo observamos que este es precisamente $\mathbb{Z}/n_1\mathbb{Z} \times \mathbb{Z}/n_2\mathbb{Z}$. □

Evidentemente, por inducción esta construcción se puede extender a cualquier cantidad finita de polinomios. Por tanto la pregunta es, ¿podemos obtener polinomios que

verifiquen las condiciones requeridas para así llevar a cabo nuestra construcción? La respuesta es afirmativa y se debe precisamente al segundo input de nuestro programa de grupos cíclicos. La razón de ser de este input es que nos permite construir tantos polinomios con grupo de Galois $\mathbb{Z}/n\mathbb{Z}$ como deseemos, ya que de hecho sabemos que son infinitos. Lo único que tenemos que hacer es volver a ejecutar nuestro programa cambiando el segundo input por $m+1$, donde m es el segundo output de la ejecución anterior.

¿Cómo podemos garantizar que los polinomios que vamos obteniendo son irreducibles en el anillo de polinomios de la extensión anterior? En vez de generar todos los polinomios a la vez, los iremos generando uno por uno, comprobando que son irreducibles en el anillo de polinomios de la extensión anterior y generando la nueva extensión antes de buscar el siguiente. Podríamos toparnos con alguno que no fuera irreducible, pero tenemos la ventaja de que podemos generar infinitos polinomios, es decir, los polinomios que no nos valgan sencillamente los descartamos y seguimos buscando hasta encontrar uno válido, que en algún momento aparecerá pues sabemos que la extensión existe.

```
def GalInvProblemAbelianFiniteGroups(L):
   x = polygen(QQ,'x');
   if len(L)==1:
      return GalInvProblemCyclicGroup(L[0],1)
   L = reordenar(L);
   JJ=[];
   for i in srange(len(L)):
      if (L[i] in JJ)==False:
         JJ.append(L[i]);
   N=QQ;
   for i in srange(len(JJ)):
      s=1;
      l = L.count(JJ[i]);
      for j in srange(l):
         y = polygen(N,'y');
         G = GalInvProblemCyclicGroup(JJ[i],s);
         f = G[0].subs(x=y);
         while (f.is_irreducible())==False:
            s = s+1;
            G = GalInvProblemCyclicGroup(JJ[i],s);
            f = G[0].subs(x=y);
         M.<a> = NumberField(f);
         N.<b> = M.absolute_field();
   return N.defining_polynomial()
```

En el final del Apéndice B se puede encontrar una explicación detallada del funcionamiento del programa, en caso de que alguna línea del mismo desconcierte al lector.

Vamos a ver como funciona el código. Nos interesa encontrar un polinomio con grupo de Galois $\mathbb{Z}/2\mathbb{Z} \times \mathbb{Z}/2\mathbb{Z} \times \mathbb{Z}/4\mathbb{Z}$. El lector interesado en grupos considerablemente

más grandes puede ejecutar el programa para obtenerlos, ya que como comprenderá, cuanto más grande es el grupo, más largo y enrevesado es el polinomio. Si ejecutamos nuestro programa sobre la lista $[2, 2, 4]$ obtenemos

$$\begin{array}{l} x^{16} + 4x^{15} + 18x^{14} + 26x^{13} + 94x^{12} + 126x^{11} + 334x^{10} + \\ 2142x^9 + 815x^8 - 1030x^7 - 1140x^6 - 3260x^5 + \\ 61961x^4 + 60576x^3 + 68633x^2 + 12930x + 8919 \end{array},$$

y afirmamos que este polinomio tiene grupo de Galois igual a $\mathbb{Z}/2\mathbb{Z} \times \mathbb{Z}/2\mathbb{Z} \times \mathbb{Z}/4\mathbb{Z}$. El lector interesado en cerciorarse de que esta afirmación es cierta puede consultar el apéndice B, en el que se incluye un programa de comprobación, cuyo objetivo es calcular el grupo de Galois de un polinomio sabiendo que dicho grupo es abeliano finito. Por la construcción del polinomio en el programa al menos si podemos garantizar que el grupo de Galois será abeliano finito, y ejecutando el programa de comprobación en nuestro polinomio obtenemos $[2, 2, 4]$.

Finalmente, se incluye también en el Apéndice E una tabla en la que se resuelve el problema inverso de Galois para todos los grupos abelianos finitos de orden menor o igual que 15. Mediante el uso del programa se puede resolver hasta casos mucho más grandes, y el lector interesado en dichos polinomios puede hacer uso del programa para calcularlos.

CAPÍTULO 3

El problema inverso de Galois para grupos simétricos

En este capítulo vamos a tratar de dar respuesta al problema inverso de Galois trabajando con grupos simétricos. Después haremos lo mismo que en el capítulo anterior y programaremos en Sage un código que resuelva dicho problema de modo efectivo.

3.1. Existencia de un polinomio con grupo de Galois S_n

La prueba de que el problema inverso de Galois se puede resolver cuando hablamos de grupos simétricos se puede realizar de forma excepcionalmente rápida, utilizando el argumento desarrollado por B.L. van der Waerden en [8]. Esta prueba es totalmente constructiva, lo que nos ayudará luego a programarlo. Para llevarla a cabo tan solo necesitamos dos resultados previos.

Antes de empezar, recordemos el Teorema de Cayley.

Teorema 3.1.1. *Todo grupo G es isomorfo a un subgrupo de las biyecciones de G. En particular, si G es de orden n es isomorfo a un subgrupo de S_n.*

Es natural considerar el grupo de Galois de un polinomio como un subgrupo de S_n, ya que la acción del grupo sobre las raíces del polinomio consiste precisamente en permutarlas. En particular, si consideramos f un polinomio irreducible sobre $\mathbb{Q}$ de grado n, por el Teorema 1.2.7 se tiene que f es separable y no tiene raíces repetidas. Como la acción del grupo de Galois sobre las raíces de f consiste en permutarlas y no hay ninguna repetida, obtenemos que el grupo de Galois permuta las n raíces y por tanto es un subgrupo transitivo de S_n.

El siguiente resultado de Dedekind es muy útil para identificar como son las permutaciones del grupo de Galois sobre $\mathbb{Q}$ de un polinomio en $\mathbb{Z}[x]$, una prueba de dicho resultado se puede encontrar en el capítulo 13 de [3].

Teorema 3.1.2. *Sea f un polinomio en $\mathbb{Z}[x]$ de grado n tal que no posee raíces múltiples y p un primo tal que la reducción de $f(x)$ a $\mathbb{F}_p[x]$ se puede descomponer como*

$$\tilde{f}(x) \equiv g_1(x)g_2(x)\cdots g_l(x) \in \mathbb{F}_p[x],$$

donde cada $g_i(x)$ son polinomios mónicos irreducibles distintos entre sí. Si llamamos δ_i al grado de $g_i(x)$, entonces el grupo de Galois de $f(x)$ sobre $\mathbb{Q}$ visto como subgrupo de S_n posee una permutación del tipo $(\delta_1, \delta_2, \ldots, \delta_l)$, es decir, una composición de δ_i-ciclos.

Proposición 3.1.3. *Dados p un primo y $n \in \mathbb{N}$, existe un polinomio de grado n que es mónico e irreducible en $\mathbb{F}_p[x]$*

Demostración. El caso en el que $n = 1$ es trivial, pues cualquier polinomio lineal es irreducible. Por lo que podemos suponer que $n > 1$.

Consideramos el polinomio $f(x) = x^{p^n} - x \in \mathbb{F}_p[x]$ y sea L su cuerpo de descomposición. Denotamos también por R el conjunto de raíces de f. Es claro que $R \subseteq L$.

Observamos ahora que 0 es una raíz de f por lo que podemos escribir f como $x(x^{p^n-1} - 1)$. Consideramos el polinomio $x^{p^n-1} - 1$, y observamos que su derivada es $(p^n - 1)x^{p^n-2}$, que es evidentemente coprimo con $x^{p^n-1} - 1$ por cuanto su único factor es x que aparece de manera múltiple y x no es factor de $x^{p^n-1} - 1$ pues 0 no es raíz de este polinomio. Así que por el contrarrecíproco de la Proposición 1.2.2 tenemos que $x^{p^n-1} - 1$ no tiene raíces repetidas[1]. Por tanto, se tiene que R tiene p^n elementos.

Es fácil probar que R es un cuerpo[2], lo que sumado a que contiene todas las raíces de f fuerza la identidad $L = R$. Tenemos que L es un cuerpo finito y por tanto su grupo multiplicativo es cíclico, digamos $L^* = \langle \alpha \rangle$. Con esta notación, obtenemos que $L = \mathbb{F}_p(\alpha)$, una extensión de un cuerpo finito con p^n elementos. Necesariamente $[L : \mathbb{F}_p] = n$ y por tanto $Irr_{\mathbb{F}_p}(\alpha)$ es un polinomio irreducible de grado n sobre $\mathbb{F}_p$. □

Antes de enunciar el resultado principal de la sección, observamos que el polinomio $x^3 - 2$ tiene como grupo de Galois el grupo S_3. Este caso lo ponemos a parte porque la prueba de van der Waerden funciona a partir de $n = 4$.

Teorema 3.1.4. *(Solución del problema inverso de Galois para grupos simétricos) Sea $n \geq 4$ y S_n el grupo simétrico de orden n. Entonces existe una extensión de Galois $L/\mathbb{Q}$ tal que $Gal(L/\mathbb{Q}) \cong S_n$.*

Demostración. Vamos a utilizar la Proposición 3.1.3 para construir tres polinomios mónicos e irreducibles de grado n, f_1, f_2, f_3 sobre $\mathbb{F}_2$, $\mathbb{F}_3$, $\mathbb{F}_5$, respectivamente y que verifiquen lo siguiente:

- f_1 es irreducible de grado n en $\mathbb{F}_2[x]$.
- Construimos g_1 un polinomio irreducible de grado $n-1$ sobre $\mathbb{F}_3$ y g_2 un polinomio lineal sobre $\mathbb{F}_3$. f_2 será igual a $g_1 g_2$.

[1] Notar que no podíamos aplicar esta proposición directamente al polinomio f pues no está dentro de las hipótesis de la Proposición 1.2.2.

[2] Se tiene que $0, 1 \in R$. Además, $a \in R$ sí y solamente si $a^{p^n} - a = 0$, es decir, si $a^{p^n} = a$. Así, si $a, b \in R$ y $a \neq 0$ se tiene que $(ab)^{p^n} = a^{p^n} b^{p^n} = ab$ y $(a^{-1})^{p^n} = (a^{p^n})^{-1} = a^{-1}$, y además $(a+b)^{p^n} \equiv a^{p^n} + b^{p^n} \pmod{p}$. Así que $a + b, ab, a^{-1} \in R$.

- Construimos h_1 de grado 2 irreducible sobre $\mathbb{F}_5$. Si n es impar construimos un polinomio irreducible de grado $n-2$ sobre $\mathbb{F}_5$, digamos h_2. Si n es par construimos un polinomio lineal h_2, y h_3 irreducible de grado $n-3$ sobre $\mathbb{F}_5$. En cualquier caso, f_3 es el producto de los polinomios construidos.

Vamos a construir ahora $f \in \mathbb{Z}[x]$ que sea congruente con cada f_i cuando lo reducimos al correspondiente $\mathbb{F}_p[x]$. Por el Teorema Chino de los Restos podemos conseguir esto si escribimos

$$f = -15f_1 + 10f_2 + 6f_3.$$

Observar que el coeficiente que acompaña a f_1 es -15 y no 15, esto se debe a que no cambia[3] el polinomio al reducir módulo 30, y tomando -15 en lugar de 15 aseguramos que f es un polinomio mónico. Vamos a ver que el grupo de Galois de f es isomorfo a S_n.

Llamemos L al cuerpo de descomposición de f sobre $\mathbb{Q}$ y G a su grupo de Galois. Como f es irreducible al reducirlo a $\mathbb{F}_2[x]$, es irreducible en $\mathbb{Q}[x]$, G es isomorfo a un subgrupo transitivo de S_n.

Tenemos que $f \equiv g_1 g_2 \pmod 3$. Como g_1 es irreducible, no tiene raíces múltiples y además es coprimo con g_2. Así, f no tiene raíces múltiples sobre $\mathbb{F}_3$ y por tanto estamos en las hipótesis del Teorema 3.1.2, del que seguimos que G posee un $(n-1)$-ciclo. Análogamente, reduciendo a módulo 5, obtendríamos que G posee una permutación del tipo $(n-2, 2)$, si n es impar, o bien una permutación del tipo $(n-3, 2)$, si n es par. En ambos casos, elevando esta permutación a $n-2$ o $n-3$ respectivamente, obtenemos debido a que ambos números son impares, que G posee una transposición.

Supongamos pues que los ciclos $(1, 2, \cdots, n-1)$ y (i, j) pertenecen a G. Nuestro objetivo es ver que en G tenemos una transposición del tipo (k, n) para algún k, podemos suponer sin pérdida de generalidad que $n > j > i$ y ponemos $s = n - j$. Observamos que $(1, 2, \ldots, n-1)(i, j)(1, 2, \ldots, n-1)^{-1} = (i+1, j+1)$ y repitiendo esto s veces obtenemos $(i+s, n) \in G$. Ahora bien, tenemos que al conjugar una trasposición tipo (i, n) con el ciclo $(1, 2, \ldots, n-1)$ obtenemos $(i+1, n)$, mientras que al conjugarlo con su inverso obtenemos $(i-1, n)$. En resumen, hemos visto que $(i, n) \in G$ para todo $i \in \{1, 2, \ldots, n-1\}$ y por tanto cualquier trasposición está en G pues $(i, j) = (i, n)(j, n)(i, n)$. Así, dado que S_n está generado por las trasposiciones obtenemos $G \cong S_n$. □

Una observación interesante es que en este caso, a diferencia de los grupos abelianos finitos, el polinomio con grupo de Galois S_n no es el polinomio mínimo de la extensión que tiene ese grupo de Galois. Y en nuestro caso el hecho de haber obtenido el polinomio que tiene grupo de Galois S_n en vez del polinomio que genera una extensión con grupo de Galois S_n es una verdadera fortuna, ya que trabajamos con polinomios de grado n en lugar de polinomios de grado $n!$.

[3] Recordar que el polinomio se construye mediante el Teorema Chino de los Restos en $\mathbb{Z}/30\mathbb{Z}[x]$.

3.2. Cálculo de un polinomio con grupo de Galois S_n

Al igual que en el caso de los grupos abelianos finitos, vamos a tratar de usar el programa Sage para resolver de forma eficiente el problema inverso de Galois para grupos simétricos. A diferencia de dicho caso, el programa es sencillamente programar la demostración del Teorema 3.1.4 y no necesita de ningún programa adicional.

Vamos a copiar el algoritmo descrito durante la prueba, ya que la única dificultad consiste en hallar polinomios irreducibles en cuerpos finitos. Para ello utilizamos el algoritmo propuesto por la Proposición 3.1.3.

```
def polirr(p,n):
   R.<x> = PolynomialRing(GF(p));
   f  = x^(p^n)-x;
   M.<a> = f.splitting_field();
   g = M.polynomial().subs(a=x);
   return g


def ProbInvGaloisSymmetric(n):
   x = polygen(QQ,'x');
   if n==3:
      return x^3-2;
   f1 = polirr(2,n)
   g1 = polirr(3,n-1); g2 = polirr(3,1);
   f2 = g1*g2;
   h1 = polirr(5,2);
   if (n)%2==1:
      h2 = polirr(5,n-2);
      f3 = h1*h2;
   else:
      h2 = polirr(5,n-3); h3 = polirr(5,1);
      f3 = h1*h2*h3;
   f1 = f1.change_ring(QQ);
   f2 = f2.change_ring(QQ);
   f3 = f3.change_ring(QQ);
   f = -15*f1 + 10*f2 + 6*f3;
   return f
```

El programa es, como hemos dicho en un principio, un calco del algoritmo descrito en la demostración del Teorema 3.1.4. Las únicas líneas que quiza requieran una mínima explicación son las correspondientes a la instrucción `change_ring()`. Esta funcionalidad de Sage toma un polinomio en un anillo $A[x]$ y lo manda a una copia de dicho polinomio en el anillo $B[x]$, siempre y cuando este cambio tenga sentido. En este caso, mandamos polinomios en $\mathbb{F}_2[x], \mathbb{F}_3[x]$ y $\mathbb{F}_5[x]$ a polinomios en $\mathbb{Q}[x]$. Esta reescritura de los polinomios es necesaria para poder realizar operaciones entre ellos.

Este programa, a pesar de funcionar para cualquier grupo simétrico, puede resultar algo lento cuando consideramos un S_n grande, ya que el cálculo de polinomios irreducibles sobre $\mathbb{F}_p$ requiere del cálculo de un cuerpo de descomposición, y por tanto puede llevar un tiempo. En el Apéndice D ofreceremos un programa alternativo que resuelva el problema de una forma más eficiente, pero para desarrolarlo necesitamos los conceptos teóricos del siguiente capítulo.

Al igual que en el caso de los grupos abelianos finitos, el lector puede encontrar la resolución del problema inverso de Galois para los primeros grupos simétricos en el Apéndice F.

CAPÍTULO 4

El problema inverso de Galois para grupos alternados

En este capítulo vamos a abordar el problema de construir polinomios en $\mathbb{Q}[x]$ cuyo grupo de Galois sea igual a A_n. Una diferencia notable con respecto a los anteriores capítulos es que la construcción no va a ser directamente sobre $\mathbb{Q}$ si no sobre $\mathbb{Q}(t)$, para t una indeterminada. Así construiremos una familia de polinomios en $\mathbb{Q}(t)[x]$ con grupo de Galois A_n sobre $\mathbb{Q}(t)$, y gracias a un mágnifico teorema debido a Hilbert podremos concluir que existen polinomios en $\mathbb{Q}[x]$ con grupo de Galois A_n.

Para ello deberemos construir, bajo la misma idea, una familia de polinomios en $\mathbb{Q}(t)[x]$ con grupo de Galois S_n, lo que nos dará una nueva forma de resolver el problema inverso de Galois en el grupo simétrico. Un inciso importante antes de empezar es que esto no desestima la construcción del Teorema 3.1.4, ya que aquella era una construcción mucho más sencilla a nivel teórico y, además, los polinomios obtenidos mediante el Teorema 3.1.4 no son posibles de construir con nuestro nuevo método.

4.1. Preliminares

En esta sección presentaremos algunas definiciones y resultados que no han aparecido hasta ahora, pero que son fundamentales para nuestro trabajo en este capítulo.

Definición 4.1.1. *Decimos que un subgrupo $G \leq S_n$ es doblemente transitivo si dados $i, i', j, j' \in \{1, 2, \ldots, n\}$ con $i \neq i'$ y $j \neq j'$, existe un elemento $g \in G$ tal que $g(i) = i'$ y $g(j) = j'$.*

Proposición 4.1.2. *Sea $G \leq S_n$ que es doblemente transitivo y contiene una transposición. Entonces $G = S_n$.*

Demostración. Sea $(ij) \in G$ con $i, j \in \{1, 2, \ldots, n\}$. Sean $k, l\{1, 2, \ldots, n\}$ con $i \neq k$ y $j \neq l$. Como G es doblemente transitivo, para algún $g \in G$ se tiene que $g(i) = k$ y $g(j) = l$. Entonces, la conjugación por g de (i, j) es precisamente (k, l) y por ser G subgrupo se tiene $(k, l) \in G$. Así, G contiene a todas las transposiciones y debe ser igual a S_n. □

El siguiente resultado se debe a Hilbert y se incluye aquí sin prueba, debido a que la complejidad de la misma excede los límites de este trabajo.

Teorema 4.1.3. *(Teorema de Irreducibilidad de Hilbert)*

Sea $f(x, t_1, t_2, \ldots, t_n) \in \mathbb{Q}(t_1, t_2, \ldots, t_n)[x]$ un polinomio irreducible. Entonces existen infinitas tuplas $(q_1, q_2, \ldots, q_n) \in \mathbb{Q}^n$ tales que $f(x, q_1, q_2, \ldots, q_n) \in \mathbb{Q}[x]$ es irreducible y además se tiene que

$$Gal_{\mathbb{Q}(t_1,t_2,\ldots,t_n)}(f(x, t_1, t_2, \ldots, t_n)) \cong Gal_{\mathbb{Q}}(f(x, q_1, q_2, \ldots, q_n)).$$

Dado un polinomio $f(x, t_1, t_2, \ldots, t_n) \in \mathbb{Q}(t_1, t_2, \ldots, t_n)[x]$ como en el Teorema, llamaremos al polinomio $f(x, a_1, a_2, \ldots, a_n) \in \mathbb{Q}[x]$ su especialización en $(a_1, a_2, \ldots, a_n)$.

El siguiente concepto será fundamental a lo largo del capítulo. Tanto la definición como los dos resultados que se muestran a continuación han sido tomados de [2].

Definición 4.1.4. *Sea K un cuerpo y $f(t) \in K[t]$ un polinomio de grado n cuya factorización en su cuerpo de descomposición es*

$$f(t) = c(t - r_1)(t - r_2)\ldots(t - r_n).$$

Se define el discriminante de f como

$$disc(f) = \prod_{i<j}(r_i - r_j)^2.$$

Una observación interesante es que $disc(f) \in K$. En efecto, si f es separable, se tiene que $disc(f) \neq 0$ y de hecho es un polinomio simétrico en las raíces r_i de f, luego debe quedar fijo por el grupo de Galois de f, lo que implica que está en K. Si f no es separable, entonces $disc(f) = 0$ y claramente está en K.

Proposición 4.1.5. *Sea $f(t) \in K[t]$ tal que $f(t) = x^n + ax + b$. Se tiene que*

$$disc(f) = (-1)^{\frac{n(n-1)}{2}}((-1)^{n-1}(n-1)^{n-1}a^n + n^n b^{n-1}).$$

Teorema 4.1.6. *Sea $f(t) \in \mathbb{Q}[t]$ un polinomio separable de grado n. Sabemos que podemos considerar el grupo de Galois de $f(t)$ como un subgrupo de S_n ya que sus elementos permutan las n raíces de $f(t)$. Se tiene que este subgrupo es un subgrupo de A_n sí y solamente sí el discriminante de $f(t)$ es un cuadrado perfecto en $\mathbb{Q}$.*

Demostración. Sean $r_1, \ldots, r_n$ las raíces de $f(t)$ en su cuerpo de descomposición $\mathbb{Q}(r_1, r_2, \ldots, r_n)$. Consideramos $\delta(f) = \prod_{i<j}(r_i - r_j) \neq 0$ debido a que el polinomio es separable. Tenemos que $\delta^2 = disc(f)$, así que $disc(f)$ es un cuadrado perfecto en $\mathbb{Q}$ sí y solamente sí $\delta \in \mathbb{Q}$.

Consideramos $\sigma \in Gal(\mathbb{Q}(r_1, r_2, \ldots, r_n)/\mathbb{Q})$ que es el grupo de Galois de $f(t)$ y sea $sign(\sigma)$ su signatura, considerando σ como elemento en S_n. Se tiene

$$\sigma(\delta) = \prod_{i<j}(\sigma(r_i) - \sigma(r_j)) = sign(\sigma)\prod_{i<j}(r_i - r_j) = sign(\sigma)\delta.$$

La segunda igualdad se debe precisamente a que σ permuta las raíces de $f(t)$ y la signatura se puede definir como $(-1)^s$ donde s es el número de pares (r_i, r_j) que invierte[1] σ. Así como $\delta \neq 0$ se tiene que $\sigma(\delta) = \pm\delta$.

Finalmente $\sigma \in A_n$ sí y solamente sí $sign(\sigma) = 1$ sí y solamente sí $\sigma(\delta) = \delta$. De aquí concluimos que el grupo de Galois de $f(t)$ es un subgrupo de A_n sí y solamente sí fija δ, esto es, $\delta \in \mathbb{Q}$.

□

La misma demostración es válida para cualquier extensión de $\mathbb{Q}$ y en la práctica, como nos interesa trabajar sobre $\mathbb{Q}$ y extensiones de $\mathbb{Q}$, podemos aplicar el Teorema 4.1.6 a cualquier polinomio irreducible, ya que gracias al Teorema 1.2.7 todos estos polinomios son separables.

4.2. Existencia de polinomios con grupo de Galois A_n

Como mencionamos al principio del capítulo, vamos a comenzar construyendo una familia de polinomios con grupo de Galois S_n sobre $\mathbb{Q}(t)$. Es más, adelantandonos a lo que viene a continuación afirmamos que las familias

$$\begin{aligned} f(x,s) &= x^n - sx - s \in \mathbb{Q}(s)[x], \\ g(y,t) &= y^n - nty + (n-1)t \in \mathbb{Q}(t)[y], \end{aligned} \tag{4.1}$$

tienen grupo de Galois igual a S_n. Por el Teorema 4.1.3 existen infinitos $q \in \mathbb{Q}$ tal que la especialización de estas familias en q tienen grupo de Galois S_n sobre $\mathbb{Q}$. En realidad, las familias f, g son la misma salvo un cambio de variable. En lo que sigue $n \geq 3$, pues los grupos S_1 y S_2 son grupos cíclicos y no presentan interés.

El siguiente resultado será la piedra angular de nuestro argumento a la hora de encontrar una familia de polinomios con grupo de Galois A_n. La demostración se incluye salvo dos detalles cuya complejidad harían la prueba mucho más difícil y larga. Estos detalles se pueden consultar en [5, Sección 3.3]. La prueba ha sido tomada tanto de [5] como de [6].

Teorema 4.2.1. *Sea $f(x,s) = x^n - sx - s \in \mathbb{Q}(s)[x]$. El grupo de Galois de $f(x,s)$ sobre $\mathbb{Q}(s)$ es isomorfo a S_n. En particular, por el Teorema de Irreducibilidad de Hilbert, existen infinitos $q \in \mathbb{Q}$ tales que $f(x,q) \in \mathbb{Q}[x]$ es irreducible y tiene grupo de Galois S_n.*

Demostración. Empezamos viendo que $f(x,s)$ es irreducible. De no ser así existirían $g(x,s), h(x,s) \in \mathbb{Q}(s)[x]$ no constantes tales que $f(x,s) = g(x,s)h(x,s)$. En particular, cualquier especialización de f se escribiría como el producto de las especializaciones de g y h. Tomando cualquier primo p, obtenemos que $f(x,p)$ es irreducible por el criterio de Einsestein y, además, $f(x,p) = g(x,p)h(x,p)$. Obtenemos así una contradicción y por tanto $f(x,s)$ es irreducible. De esto obtenemos que su cuerpo

[1] Recordemos que σ invierte (r_i, r_j) si $i < j$ y $\sigma(j) < \sigma(i)$.

de descomposición L es una extensión de Galois, ya que $\mathbb{Q}(s)$ es un cuerpo perfecto por tener característica 0 y el Teorema 1.2.7. En particular, su grupo de Galois $Gal(L/\mathbb{Q}(s))$ es un subgrupo transitivo de S_n.

Vamos a utilizar ahora la Proposición 4.1.2 para ver que $Gal(L/\mathbb{Q}(s))$ es exactamente S_n.

Podemos escribir $f(x,s)$ como $x^n - \frac{x}{2} - \frac{1}{2} - (s-\frac{1}{2})(x+1)$. Su especialización $f(x,\frac{1}{2}) = x^n - \frac{x}{2} - \frac{1}{2}$ se puede factorizar como

$$\frac{1}{2}(x-1)\underbrace{(2x^{n-1} + 2x^{n-2} + \cdots + 2x + 1)}_{h(x)}.$$

Tenemos que $h(x)$ es irreducible pues por el criterio de Einsestein $2 + 2x + 2x^2 + \cdots + 2x^{n-2} + x^{n-1}$ es irreducible y $h(x)$ no es más que la imagen de este polinomio por el homomorfismo de anillos de polinomios

$$f(a_0 + a_1x + a_2x^2 + \cdots a_nx^n) = a_n + a_{n-1}x + \cdots a_1x^{n-1} + a_0x^n = x^n f\left(\frac{1}{x}\right).$$

Debido a que esta especialización se puede factorizar de esta manera concluimos que $Gal(L/\mathbb{Q}(s))$ es doblemente transitivo (consulta [5, Sección 3.3]).

Vamos a ver ahora que $Gal(L/\mathbb{Q}(s))$ posee una transposición. Tomamos el cambio de variable $yn = (n-1)x$ para obtener

$$y^n - s\left(\frac{1-n}{n}\right)^{n-1} y - s\left(\frac{1-n}{n}\right)^n.$$

Ahora reescalando $t = s\frac{(1-n)^{n-1}}{n^n}$ obtenemos la familia

$$g(y,t) = y^n - nty + (n-1)t \in \mathbb{Q}(t)[y].$$

Observar que esta familia es la misma que aparecía en (4.1), que como prometimos es $f(x,s)$ tras dos cambios de variables, por lo que el grupo de Galois de $g(y,t)$ sobre $\mathbb{Q}(t)$ será el mismo que el de $f(x,s)$ sobre $\mathbb{Q}(s)$. Podemos escribir $g(y,t)$ como

$$y^n - y - (n-1)(y-1) + (t-1)(-ny + n - 1),$$

así que $g(y,1) = y^n - y - (n-1)(y-1)$. Es fácil ver que este polinomio posee una raíz doble ($y = 1$) y $n-2$ raíces simples. De esto deducimos que el grupo de Galois de $g(y,t)$ sobre $\mathbb{Q}(t)$ posee una transposición (consultar [5, Sección 3.3]) y por tanto también lo posee $Gal(L/\mathbb{Q}(s))$. Aplicando la Proposición 4.1.2 deducimos que $Gal(L/\mathbb{Q}(s)) \cong S_n$. □

Antes de continuar vamos a hacer un ejemplo para ver que en efecto esta construcción funciona, hallando un polinomio con grupo de Galois S_7. El ejemplo ha sido tomado de [1].

Ejemplo 4.2.2. *Consideramos el polinomio* $f(x,s) = x^7 - sx - s$ *y su especialización* $f(x) = f(x,1) = x^7 - x - 1$. *Se tiene que*

$$\begin{aligned} f(x) &\equiv x^7 + x + 1 \pmod 2, \\ f(x) &\equiv (x^2 + x + 2)(x^5 + 2x^4 + 2x^3 + 2x + 1) \pmod 3, \\ f(x) &\equiv (x+3)(x^6 + 2x^5 + 4x^4 + 3x^3 + x^2 + 2) \pmod 5. \end{aligned}$$

Por la reducción a $\mathbb{F}_2[x]$ *vemos que* $f(x)$ *es un polinomio irreducible y por tanto* $Gal_{\mathbb{Q}}(f)$ *es un subgrupo transitivo de* S_n. *Por la Proposición 4.1.5 podemos calcular el discriminante de forma sencilla y obtener* $disc(f) = -776887$ *que no es un cuadrado perfecto en* $\mathbb{Q}$. *Así que* $Gal_{\mathbb{Q}}(f)$ *no es subgrupo de* A_7. *Por el Teorema 3.1.2 sabemos gracias a la reducción módulo* 2 *y a la reducción módulo* 5 *que* $Gal_{\mathbb{Q}}(f)$ *posee un* $7-$*ciclo y una permutación del tipo* $(2,5)$, *cuya potencia quinta es una transposición, por lo que* $Gal_{\mathbb{Q}}(f)$ *posee una transposición. Por el siguiente resultado de la teoría de grupos simétricos, cuya demostración se puede encontrar en [1], concluimos que* $Gal_{\mathbb{Q}}(f)$ *es* S_7.

Lema 4.2.3. *Para* $n \geq 2$, *si un subgrupo transitivo de* S_n *posee una transposición y un* $p-$*ciclo para cierto primo* $p > \frac{n}{2}$, *entonces ese subgrupo es* S_n.

Estamos ya en condiciones de resolver el problema inverso de Galois para grupos alternados. La solución será similar a la del Teorema 4.2.1, es decir, construiremos una familia de polinomios $g(y,u) \in \mathbb{Q}(u)[y]$ cuyo grupo de Galois será A_n y de nuevo por el Teorema 4.1.3 tendremos infinitas especializaciones sobre $\mathbb{Q}$ que preservan ese grupo de Galois. La demostración ha sido realizada en base a un esquema que se puede encontrar en [6].

Teorema 4.2.4. *Sea* $g(y,t) = y^n - nty + (n-1)t \in \mathbb{Q}(t)[y]$. *Consideremos el cambio de variables*

$$t = \begin{cases} 1 - (-1)^{\frac{n(n-1)}{2}} nu^2 & si \quad n \equiv 1 \pmod 2, \\ \dfrac{1}{1+(-1)^{\frac{n(n-1)}{2}}(n-1)u^2} & si \quad n \equiv 0 \pmod 2. \end{cases}$$

El grupo de Galois de $g(y,u)$ *sobre* $\mathbb{Q}(u)$ *es* A_n. *En particular, por el Teorema de Irreducibilidad de Hilbert, existen infinitos* $q \in \mathbb{Q}$ *tales que* $Gal_{\mathbb{Q}}(g(y,q)) \cong A_n$.

Demostración. Por la demostración del Teorema 4.2.1, la familia de polinomios $g(y,t)$ tiene grupo de Galois S_n sobre $\mathbb{Q}(t)$. Llamemos L al cuerpo de descomposición de $g(y,t)$ sobre $\mathbb{Q}(t)$. Por la Proposición 4.1.5 sabemos que $disc(f) = (-1)^{\frac{n(n-1)}{2}} n^n (n-1)^{n-1} t^{n-1}(1-t)$. El objetivo del cambio de variables descrito en el enunciado es conseguir un polinomio cuyo discriminante sea un cuadrado perfecto.

Si n es impar, tras el cambio de variable obtenemos que $disc(g) = n^{n+1}(n-1)^{n-1}t^{n-1}u^2$ es un cuadrado perfecto en $\mathbb{Q}(u)$. Si n es par, $disc(g) = n^n(n-1)^n t^n u^2$, que es un cuadrado perfecto en $\mathbb{Q}(u)$.

En ambos casos es claro que $\mathbb{Q}(t) = \mathbb{Q}(u^2)$ y por tanto si demostramos que $\mathbb{Q}(u) \subset L$ tiene sentido considerar el grupo de Galois $Gal(L/\mathbb{Q}(u))$ como grupo de Galois de g sobre $\mathbb{Q}(u)$, ya que en este caso la extensión sería de Galois por ser $L/\mathbb{Q}(u^2)$ una extensión de Galois finita y $\mathbb{Q}(u)$ un cuerpo intermedio. Entonces, si tiene sentido

hablar de grupo de Galois de g sobre $\mathbb{Q}(u)$ sabemos que esté debe ser un subgrupo de A_n, por ser $disc(g)$ un cuadrado perfecto en $\mathbb{Q}(u)$ y aplicando el Teorema 4.1.6.

Veamos, vamos a considerar $\delta = \sqrt{disc(g)}$ como en la demostración del Teorema 4.1.6. Tenemos que es una expresión en función de las raíces de g, por lo que $\delta \in L$. Así que $\mathbb{Q}(u^2, \delta) \subset L$. Por otra parte, $\mathbb{Q}(u^2, \delta) = \mathbb{Q}(u)$ ya que

$$\delta = \begin{cases} \sqrt{n^{n+1}(n-1)^{n-1}t^{n-1}}u & si \quad n \equiv 1 \pmod 2, \\ \sqrt{n^n(n-1)^n t^n}u & si \quad n \equiv 0 \pmod 2. \end{cases}$$

Y recordemos que t se puede expresar en términos de u^2 y viceversa.

Así hemos obtenido que $L/\mathbb{Q}(u)$ es una extensión de Galois y $Gal(L/\mathbb{Q}(u)) \leq A_n$. Ahora, basta aplicar el Teorema Fundamental de la Teoría de Galois para obtener

$$n! = [L : \mathbb{Q}(u^2)],$$

debido a que $Gal_{\mathbb{Q}(t)}(g(y,t)) = S_n$ por el Teorema 4.2.1 y $\mathbb{Q}(t) = \mathbb{Q}(u^2)$. Por tanto,

$$n! = [L : \mathbb{Q}(u)][\mathbb{Q}(u) : \mathbb{Q}(u^2)] = 2[L : \mathbb{Q}(u)].$$

De donde obtenemos, de nuevo por el Teorema Fundamental de la Teoría de Galois, que $Gal(L/\mathbb{Q}(u))$ tiene orden $\frac{n!}{2}$, por lo que debe ser $Gal(L/\mathbb{Q}(u)) \cong A_n$. □

Una observación a raíz de la demostración es que para infinitos t racionales, se tiene que $g(y,t) = y^n - nty + (n-1)t \in \mathbb{Q}[y]$ tiene como grupo de Galois A_n. En efecto, la familia $g(y,u)$ construida en el Teorema 4.2.4 no es más que $g(y,t)$ tras un cambio de variable, por lo que a cada $u \in \mathbb{Q}$ tal que $Gal_{\mathbb{Q}}(g(y,u)) \cong A_n$ le corresponde un $t \in \mathbb{Q}$ tal que $Gal_{\mathbb{Q}}(g(y,t)) \cong A_n$. Así que la familia $g(y,t)$ descrita en (4.1) tiene la propiedad de que para infinitos racionales t tiene grupo de Galois sobre $\mathbb{Q}$ igual a S_n y para otros infinitos t tiene grupo A_n.

Otra observación de la demostración es que una vez que construimos la extensión $L/\mathbb{Q}(u)$, hablamos de grupo de Galois de g sobre $\mathbb{Q}(u)$, ya que el cuerpo de descomposición sigue siendo el mismo. Sin embargo, $g(u)$ no tiene porque ser necesariamente irreducible sobre $\mathbb{Q}(u)$, lo cual no constituye un problema ya que sí podemos garantizar que $L/\mathbb{Q}(u)$ es una extensión de Galois, ya que como hemos visto en la demostración $L/\mathbb{Q}(u^2)$ es de Galois finita y por tanto para cualquier cuerpo K intermedio se tiene que L/K es de Galois.

Terminamos con un ejemplo en el que mostramos un polinomio en $\mathbb{Q}[x]$ con grupo de Galois igual a A_9.

Ejemplo 4.2.5. *Consideramos el polinomio $f(x) = x^9 + 72x - 64$, que es la especialización de la familia $g(x,u)$ presentada en el Teorema 4.2.4 en $u = 1$, o equivalentemente, para ilustrar nuestra observación anterior, es la especialización de $g(x,t) = x^9 - 9tx + 8t$ en el t obtenido al deshacer el cambio de variable, más concretamente, $t = 8$.*

Sabemos que $g(x,u)$ tiene grupo de Galois A_9 sobre $\mathbb{Q}(u)$ y para infinitos $u \in \mathbb{Q}$ este grupo se conserva sobre $\mathbb{Q}$. Queremos ver que $u = 1$ es uno de esos infinitos racionales. Para lograrlo vamos a llevar a cabo tres comprobaciones: primero debemos ver

que el polinomio es irreducible, lo que garantizará que su grupo de Galois es transitivo; después demostraremos que $Gal_{\mathbb{Q}}(f)$ posee un $3-ciclo$ y un $7-ciclo$ y gracias al siguiente resultado de la teoría de grupos simétricos, cuya demostración se puede encontrar en [1], podremos afirmar que $Gal_{\mathbb{Q}}(f) \cong A_9, S_9$.

Lema 4.2.6. *Sea $n \geq 3$ y G un subgrupo transitivo de S_n. Sea p un primo con $p > \frac{n}{2}$, si G posee un $3-ciclo$ y un $p-ciclo$, entonces G es bien S_n, bien A_n.*

Después, calcularemos el discriminante de f mediante la fórmula provista por la Proposición 4.1.5 y veremos que es un cuadrado perfecto.

Primero observamos que el polinomio es irreducible ya que su reducción a $\mathbb{F}_{11}[x]$ es irreducible. Ahora, reduciendo el polinomio a $\mathbb{F}_{47}[x]$ obtenemos

$$f(x) \equiv (x+23)(x^3+21x^2+x+25)(x^5+3x^4+42x^3+36x^2+23x+7) \pmod{47}.$$

Gracias al Teorema 3.1.2, sabemos que el grupo de Galois de f sobre $\mathbb{Q}$ posee una permutación de la forma $(3,5)$ y elevando dicha permutación a 5 y 3 obtenemos que el grupo de Galois de f sobre $\mathbb{Q}$ posee un $3-ciclo$ y un $5-ciclo$, y por el Lema 4.2.6 obtenemos $Gal_{\mathbb{Q}}(f) \cong S_9, A_9$. Se tiene además que $disc(f) = 990677827584^2$ y por tanto $Gal_{\mathbb{Q}}(f) \cong A_9$.

4.3. Cálculo de polinomios con grupo de Galois A_n

En la sección anterior demostramos que la familia de polinomios $g(y,t) = y^n - nty + (n-1)t \in \mathbb{Q}(t)[y]$ tenía grupo de Galois A_n sobre $\mathbb{Q}(t)$, y gracias al Teorema de Irreducibilidad de Hilbert obteniamos que para infinitos racionales $t \in \mathbb{Q}$, $Gal_{\mathbb{Q}}(g(y,t)) \cong A_n$. La pregunta es, ¿cómo identificamos dichos racionales de forma eficiente?

En el caso en el que una especialización entera de la familia $g(y,t)$ posea grupo de Galois A_n, es claro que podemos encontrarla repitiendo las comprobaciones del Ejemplo 4.2.5. Estas comprobaciones se pueden realizar de forma muy eficiente con Sage, y en eso nos vamos a centrar durante esta sección. Al igual que en el caso de los capítulos anteriores, vamos a diseñar un código Sage que encuentre un polinomio con el $n-$ésimo grupo alternado como grupo de Galois.

La diferencia con los capítulos anteriores es que en este caso no tenemos una demostración constructiva de polinomios sobre $\mathbb{Q}$ que programar, si no que partimos de que sabemos como debe ser el polinomio y debemos comprobar que la especialización elegida es, en efecto, la que buscamos.

Como ya hemos dicho, una forma eficiente de buscar dicho polinomio es programar el Teorema 3.1.2 y aplicarselo a especializaciones enteras de la familia $g(y,t)$. Por desgracia, no siempre existe una especialización entera de $g(y,t)$ que tenga grupo de Galois A_n, como ilustra el siguiente resultado.

Proposición 4.3.1. *Sea $n \in \mathbb{N}$ un natural tal que $n \equiv 0 \pmod 4$. No existe ninguna especialización entera de $g(y,t)$ con grupo de Galois $Gal_{\mathbb{Q}}(g(y,t)) \cong A_n$.*

Demostración. Vamos a calcular el discriminante del polinomio $g(y,t) = y^n - nty + (n-1)t$ sabiendo que $n \equiv 0 \pmod 4$. Por la Proposición 4.1.5 sabemos que

$$disc(g) = -(n-1)^{n-1}n^n t^n + n^n(n-1)^{n-1}t^{n-1} = n^n(n-1)^{n-1}t^{n-1}(1-t).$$

Como n es par, podemos despejar cuadrados y obtener que $disc(g)$ es un cuadrado si y solamente si $(n-1)t(1-t)$ es un cuadrado. En particular, una condición necesaria para que $disc(g)$ sea un cuadrado es que $t \in [0,1]$.

Pero para los valores $t = 0, 1$, obtenemos que el discriminante es 0 y por tanto el polinomio posee al menos una raíz múltiple, es decir, no es un polinomio separable y su cuerpo de descomposición no es una extensión de Galois. □

Así que si $n \equiv 0 \pmod 4$, entonces ninguna especialización entera de $g(y,t)$ tiene grupo de Galois A_n.

En lugar de estudiar que condiciones debe verificar n para que $g(y,t)$ admita una especialización entera con grupo de Galois A_n, vamos a intentar abordar el problema de otro modo. Primero, vamos a hallar una fórmula que a cada racional m le asigne un racional t tal que el discriminante de $g(y,t)$ sea un cuadrado perfecto.

Proposición 4.3.2. *Sea $g(y,t) = y^n - nty + (n-1)t$ y sea $m \in \mathbb{Q}$. Si se tiene que*

$$t = \begin{cases} \frac{n-1}{n-1+m^2} & si \quad n \equiv 0 \pmod 4, \\ \frac{m^2-n}{m^2} & si \quad n \equiv 1 \pmod 4, \\ \frac{n-1}{n-1-m^2} & si \quad n \equiv 2 \pmod 4, \\ \frac{m^2+n}{m^2} & si \quad n \equiv 3 \pmod 4. \end{cases}$$

Entonces el discriminante de $g(y,t)$ es un cuadrado perfecto en $\mathbb{Q}$.

Demostración. Vamos a demostrar el caso $n \equiv 0 \pmod 4$. La prueba de los otros tres casos requiere solo de una ligera modificación del argumento.

Por la demostración de la Proposición 4.3.1, sabemos que el discriminante de $g(y,t)$ es un cuadrado perfecto sí y solamente sí $(n-1)t(1-t)$ es un cuadrado perfecto.

Consideremos la cónica dada por $z^2 = (n-1)t(1-t)$, cuyas soluciones racionales son precisamente los puntos en los cuales el discriminante de $g(y,t)$ es un cuadrado perfecto en $\mathbb{Q}$.

Observamos que la cónica pasa por el punto $(0,0)$ y pongamos $z = mt$ para $m \in \mathbb{Q}$. Entonces, reescribiendo la ecuación de la cónica obtenemos la ecuación

$$m^2t^2 = (n-1)t(t-1). \tag{4.2}$$

Las soluciones t de esta ecuación son precisamente los puntos de intersección de la cónica con la recta que pasa por el origen y tiene pendiente m, en particular, si t es una solución racional de la ecuación, entonces (t, mt) es un punto de la cónica con coordenadas racionales y en particular, t es tal que el discriminante de $g(y,t)$ es un cuadrado perfecto en $\mathbb{Q}$.

Así que, debemos encontrar las soluciones de 4.2, que son claramente $t = 0, \frac{n-1}{n-1+m^2}$. □

Así que ya sabemos encontrar especializaciones racionales de la familia $g(y,t)$ con discriminante un cuadrado perfecto en $\mathbb{Q}$. Pero, ¿cómo podemos garantizar que alguna de estas especializaciones tenga grupo de Galois A_n? Si no trabajamos con polinomios en $\mathbb{Z}[x]$ entonces el Teorema 3.1.2 no es aplicable. Afortunadamente, gracias al siguiente resultado podemos aplicar el Teorema 3.1.2 a cualquier especialización racional $g(y,t)$, pasando antes por un polinomio intermedio.

Proposición 4.3.3. *El grupo de Galois sobre $\mathbb{Q}$ del polinomio $g(y,\frac{a}{b}) = y^n - \frac{na}{b}y + \frac{(n-1)a}{b}$ es el grupo de Galois sobre $\mathbb{Q}$ del polinomio $h(x) = x^n - nab^{n-2}x + (n-1)ab^{n-1}$.*

Demostración. Sea L el cuerpo de descomposición de $g(y,\frac{a}{b})$, que es una extensión de Galois por ser un cuerpo de descomposición sobre $\mathbb{Q}$. Consideremos el polinomio $b^n g(y,\frac{a}{b}) = b^n y^n - nab^{n-1}y + (n-1)ab^{n-1}$ y realicemos el cambio de variables $x = by$. Entonces obtenemos que $b^n g(y,\frac{a}{b}) = h(x)$ y por tanto el cuerpo de descomposición de h sobre $\mathbb{Q}$ es L. Se sigue que

$$Gal_{\mathbb{Q}}(h(x)) = Gal(L/\mathbb{Q}) = Gal_{\mathbb{Q}}\left(g\left(y,\frac{a}{b}\right)\right).$$

□

Después de estos resultados, podemos escribir un programa Sage que calcule un polinomio con grupo de Galois A_n.

```
def ProbInvGalois_AlternatingGroup(n,ml,numerodeprimos):
   x = polygen(QQ,'x');
   primos = primes_first_n(numerodeprimos);
   while 1!=0:
      m=0;
      while m==0:
         m = randint(1,ml)/randint(1,ml);
      if n%4==0:
         a = n-1;
         b = n-1+m^2;
      elif n%4==1:
         a = m^2-n;
         b = m^2;
      elif n%4==2:
         a = n-1;
         b = n-1-m^2;
      else:
         a = n+m^2;
         b = m^2;
      t = a/b;
      g = x^n - n*t*x + (n-1)*t;
      if (g.is_irreducible)==True:
         if (t in ZZ)==True:
```

```
            if (comprobador(n,[(n-1)*t,-n*t],primos))==True:
               returng g
         if (t in ZZ) == False:
            if (comprobador(n,[(n-1)*a*(b^(n-1)),-n*a*(b^(n-2))],primos))
            ==True:
               return g
```

Vemos que el programa recibe como inputs n, un valor que servirá para generar un $m \in \mathbb{Q} \setminus \{0\}$ y una tolerancia de primos hasta los que estamos dispuestos a reducirnos en nuestras comprobaciones equivalentes a las del Ejemplo 4.2.5. Una observación: el programa devolverá siempre un polinomio de la familia $g(y,t)$ con grupo de Galois A_n, pero para cada uno de estos polinomios hemos encontrado un polinomio en $\mathbb{Z}[x]$ con grupo de Galois A_n. Así que a partir de cada polinomio podemos obtener gracias a la Proposición 4.3.3 nuevos polinomios con nuestro grupo de Galois. Es más, la generación del $m \in \mathbb{Q}$ que utilizaremos en el programa es aleatoria, así que cada vez que ejecutemos el programa obtendremos un polinomio distinto, aunque siempre válido.

Como en el caso de los grupos abelianos finitos, el programa bebe de subrutinas que no se incluyen en esta sección por cuestiones de espacio. El lector interesado puede consultar estas subrutinas en el Apéndice C. El funcionamiento del programa es sencillo, primero calcula un $t \in \mathbb{Q}$ apropiado mediante la Proposición 4.3.2, despues comprueba que nuestro polinomio es irreducible y llama a la subrutina `comprobador` para determinar si nuestro polinomio es válido o no. El funcionamiento de esta subrutina consiste en llevar a cabo las comprobaciones equivalentes a las del Ejemplo 4.2.5 bien en nuestro polinomio `g` si este está en $\mathbb{Z}[x]$, bien en el polinomio construido mediante la Proposición 4.3.3 si `g` no tiene coeficientes enteros.

Observar que para encontrar polinomios con grupo de Galois A_n lo único que necesitamos es asignar valores `True` o `False` a ciertas variables en función de como sean los coeficientes de nuestro polinomio. Es decir, no tenemos que pasar por la construcción de los grupos de Galois en sí ni tampoco de los cuerpos de descomposición. Así el programa es muy eficiente y calcula rápidamente polinomios con grupo de Galois, por ejemplo, A_{120}. Si hubíeramos pasado por construir el cuerpo de descomposición, este tendría grado $\frac{120!}{2}$ sobre $\mathbb{Q}$, lo que requiere un esfuerzo computacional muchísimo más elevado.

Un último comentario con respecto al programa: estamos generando número aleatorios $m \in \mathbb{Q} \setminus \{0\}$ y a partir de ellos unos $t \in \mathbb{Q}$ apropiados. En el caso $n \equiv 2 \pmod 4$ podemos encontrarnos con la mala suerte de que $n-1$ sea un cuadrado perfecto y nuestro m sea precisamente su raíz cuadrada. En ese caso el programa fallará porque no podemos dividir por 0 al buscar la t. Pero no pasa nada, una nueva ejecución y obtendremos el polinomio deseado.

Al igual que en capítulos precedentes, el lector puede encontrar una tabla con polinomios cuyo grupo de Galois es A_n para los primeros casos, en el Apéndice F.

CAPÍTULO 5
Conclusiones finales

A pesar de que el problema inverso de Galois en la actualidad no está resuelto, los casos más sencillos sobre $\mathbb{Q}$ pueden ser tratados con las herramientas básicas de la teoría de Galois. A lo largo del presente texto hemos estudiado dichos casos en profundidad y hemos comprobado que las matemáticas y la computación discurren, en muchos casos, por caminos adyacentes.

En el caso de los grupos simétricos, hemos llevado a cabo una demostración constructiva que en la práctica, y especialmente en casos pequeños, puede ser realizada simplemente siguiendo el algoritmo. Sin embargo, hemos visto que las construcciones llevadas a cabo en el caso de los grupos abelianos finitos y los grupos alternados, a pesar de ser matemáticamente correctas y demostrar que el problema tiene solución, no son sencillas de llevar a la práctica si nos interesa una solución explícita.

En el caso de los grupos abelianos finitos, tenemos que trabajar con extensiones muy grandes de los racionales y saber como identificar subgrupos y subextensiones mediante Teoría de Galois. Esto no es sencillo y por tanto el atajo proporcionado por Sage para calcular la extensión es necesario. Por su parte, los grupos alternados aparecen siempre como grupo de Galois de una familia fija de polinomios, pero hemos visto que no cualquier miembro de la familia es válido. Sin ir más lejos, en el Ejemplo 4.2.5 hemos tenido que trabajar con un polinomio de grado 9 en $\mathbb{F}_{47}[x]$ para verificar que nuestro candidato era apropiado. Por ello vuelve a ser necesario el empleo de un ordenador.

En este proyecto podríamos habernos olvidado de estos aspectos computacionales y haber abordado teóricamente casos de grupos más complejos. Sin embargo, hemos creído más apropiado dar a las familias consideradas un tratamiento más profundo en vez de analizar más familias por dos razones: primero, porque el siguiente nivel de dificultad requiere unas herramientas matemáticas mucho más complejas, las curvas elípticas, y tratar estos objetos podría haber abarcado el proyecto completo; pero segundo y más importante, porque olvidarnos de la parte computacional del trabajo, solo habríamos resuelto el problema inverso de Galois a medias. Parafraseando el inicio del texto, dado un grupo G, queremos encontrar una extensión de $\mathbb{Q}$ que lo tenga como grupo de Galois, o lo que es lo mismo, un polinomio que caracterice dicha extensión.

APÉNDICE A
Demostraciones de algunos resultados

Proposición 1.2.2. *Sea $f(x) \in \mathbb{F}_p[x]$ un polinomio de grado n tal que $p \nmid na_n$ donde a_n es el coeficiente del término de mayor grado de $f(x)$. Supongamos que $f(x)$ tiene una raíz repetida en $\mathbb{F}_p[x]$, entonces $mcd(f, f') \neq 1$, donde $f'(x)$ representa la derivada del polinomio $f(x)$.*

Demostración. Suponemos que f tiene una raíz repetida, digamos α. Entonces podemos escribir $f(x)$ como $(x-\alpha)^s g(x)$ donde $g(x)$ es un polinomio coprimo con $(x-\alpha)$ y $s > 1$. Se tiene $f'(x) = s(x-\alpha)^{s-1}g(x) + (x-\alpha)^s g'(x)$. En esta igualdad se debe entender que trabajamos con polinomios en $\mathbb{F}_p[x]$ y la derivación está bien definida si consideramos el polinomio en $\mathbb{Z}[x]$, derivamos y a continuación reducimos al anillo $\mathbb{F}_p[x]$.

Si comprobamos que $f'(x)$ no es el polinomio nulo, entonces es claro que f, f' no son coprimos, pues $(x-\alpha)$ divide a ambos polinomios.

Veamos pues que $f'(x)$ no es el polinomio nulo. En primer lugar, observamos que el coeficiente del término de mayor grado de $g(x)$ es precisamente a_n, ya que $(x-\alpha)^s$ es mónico. Es decir $g(x) = a_n x^{n-s} + b_{n-s-1}x^{n-s-1} + \cdots + b_1 x + b_0$. A partir de esta observación vemos que en la definición de $f'(x)$ el término de mayor grado a la izquierda de la suma es $sa_n x^{n-1}$ y el término de mayor grado a la derecha es $(n-s)a_n x^{n-1}$, por lo que el coeficiente de grado $n-1$ de f' es na_n. Como hemos supuesto que $na_n \not\equiv 0 \pmod{p}$, seguimos que f' no es el polinomio nulo. □

Lema 2.1.1. *Sea G un grupo abeliano y sean $x_1, x_2, ...x_m \in G$ tales que $o(x_i) = n_i$ y $mcd(n_i, n_j) = 1$ si $i \neq j$. Se tiene que $o(x_1 \cdots x_m) = \prod_{i=1}^{m} n_i$.*

Demostración. Lo hacemos por inducción en m:

- Si $m = 1$, no hay nada que probar.

- Si $m = 2$, se tiene que $(x_1x_2)^{n_1n_2} = 1$ pues el grupo es abeliano, por tanto $o(x_1x_2) \mid n_1n_2$. Supongamos que $d = o(x_1x_2) < n_1n_2$, entonces se tiene que $(x_1x_2)^d = 1$ y como estamos en un grupo abeliano se tiene que bien $x_1^d = 1$, $x_2^d = 1$, bien $x_1^d \neq 1$, $x_2^d \neq 1$. El primer caso es imposible, ya que $n_1 \mid d$, $n_2 \mid d \Rightarrow n_1n_2 \mid d \Rightarrow d \geq n_1n_2$, lo que es una contradicción. Tampoco puede ser el segundo caso pues eso supondría que $(x_1^d)^{-1} = x_2^d$, lo que implica por una parte que $x_2^d \in \langle x_1^d \rangle \Rightarrow o(x_2^d) \mid o(x_1^d) \mid n_1$; por otra parte se tiene que $o(x_2^d) \mid n_2$. Por tanto, utilizando que n_1 y n_2 son coprimos se tiene que $o(x_2^d) = 1$, lo que es una contradicción ya que $x_2^d \neq 1$. Luego $d = n_1n_2$.

- Suponiendo el resultado cierto para $m - 1$ es inmediato probarlo para m, repitiendo el argumento utilizado en el caso $m = 2$. □

Lema 2.1.2. *Sea $(F, +, \cdot)$ un cuerpo y sea $(F^*, \cdot)$ su grupo multiplicativo. Sea $G \leq F^*$ finito, entonces G es cíclico.*

Demostración. Como F^* es abeliano, G es un grupo abeliano finito, y se tiene por el Teorema 1.2.1 que $G \cong \mathbb{Z}/n_1\mathbb{Z} \times \cdots \mathbb{Z}/n_s\mathbb{Z}$ donde $n_i \mid n_{i+1}\ \forall i < s$. Podemos considerar $\mathbb{Z}/n_i\mathbb{Z}$ como grupos con notación multiplicativa. Entonces para cualquier $b \in \mathbb{Z}/n_i\mathbb{Z}$ se tiene que $b^{n_i} = 1$ y por tanto $b^{n_s} = 1$. Se sigue que si $g \in G$ entonces $g^{n_s} = 1$ por lo que g es una raíz del polinomio $x^{n_s} - 1$ que tiene a lo sumo n_s raíces dentro de G. Pero como $o(G) = \prod_{i=1}^{s} n_i$, debe ser $s = 1$, lo que implica que G es cíclico. □

APÉNDICE B
Complementos para la sección 2.2

Durante todo este Apéndice se seguirá la notación de los resultados descritos en la sección 2.1. Como se dice en la sección 2.2, el comienzo de este apéndice es una explicación del código que resuelve el problema inverso de Galois para los grupos cíclicos finitos, así que comenzamos por recordar dicho programa.

```
def GalInvProblemCyclicGroup(n,s):
   x = polygen(QQ,'x');
   m = generam(n,s);
   p = (n*m)+1;
   if m==1:
      return cyclotomic_polynomial(p)
   Galp = CyclotomicField(p).galois_group();
   H=Galp;
   i=0;
   while H.order()!=m:
      H=Galp.subgroup([Galp[i]]);
      i=i+1;
   K=H.fixed_field()[0];
   return [K.defining_polynomial(),m]
```

Observamos que en los dos programas destinados a la resolución del problema inverso de Galois para grupos cíclicos hemos añadido un input que nos indica a partir de que número natural empezamos a buscar nuestra m. Como el lector puede observar, ese input no es necesario para el correcto funcionamiento del programa, no obstante, será un elemento crucial a la hora de escribir el programa que calcule el polinomio de un grupo abeliano finito cualquiera.

Aclaramos ahora algunas lineas del programa:

- La primera línea simplemente sirve para indicar a Sage que los polinomios que consideramos están en $\mathbb{Q}[x]$. Será una línea muy común en todos nuestros programas, ya que sin ella no funcionan correctamente.

- Una vez calculados m y p observamos que si $m = 1$ hemos terminado. Esto se debe a que, como vimos en la sección anterior, en este caso la extensión buscada es la ciclotómica, y Sage tiene guardado en su base de datos los polinomios ciclotómicos, por lo que no necesitamos ningún calculo extra.

- Una vez que hemos calculado m y p, y teniendo que $m \neq 1$, procedemos tal cual hemos hecho en la sección anterior. En primer lugar buscamos el grupo $Gal(\mathbb{Q}(\epsilon)/\mathbb{Q})$ donde ϵ es una raíz p-ésima primitiva de la unidad. Este grupo es el que hemos denominado `Galp`. Lo duplicamos a través de `H`, que desempeñará la misma función que el H del Teorema 2.1.6. Empezamos suponiendo que `H` es el grupo total y lo vamos variando entre los distintos subgrupos de `Galp`, aprovechando el hecho de que sabemos que existe un único subgrupo cuyo orden es el que buscamos, a saber, m. La forma de buscar estos subgrupos está en el hecho de que como `Galp` es un grupo cíclico, todos sus subgrupos son cíclicos, y por tanto nuestra única misión es recorrer los elementos del grupo formando sus grupos cíclicos hasta encontrar el buscado. Una vez que tenemos `H`, hayamos su cuerpo fijo que es la extensión que buscamos.

- Observar que al hallar el cuerpo fijo se pide que consideremos tan solo el primer elemento del output de la función de Sage que halla esta extensión. Esto se debe a que el output de esta función está compuesto por la extensión, así como por una función de inclusión de $\mathbb{Q}$ en la extensión. Solo nos interesa el primero.

- Finalmente, observamos de nuevo que el m forma parte del output de nuestra función. De nuevo, la razón de esto radica en un programa posterior.

Vamos ahora a definir un programa de comprobación, es decir, un programa tal que dado un polinomio obtenido con nuestros programas, nos devuelva el grupo abeliano finito que tiene como grupo de Galois. Sage incorpora una opción que calcula grupos de Galois y que usaremos en nuestro programa. La necesidad de definir este programa que buscamos es que Sage devuelve el grupo como un objeto que verifica las propiedades de grupo y que podemos manejar como tal, pero con el inconveniente que solo indica que ese es el grupo de Galois buscado, de una manera en la cual es difícil identificar el grupo. Mediante este objeto que Sage devuelve podemos realizar comprobaciones que nos muestren su naturaleza, y ese es el objetivo de nuestro programa de comprobación.

Vamos a aprovecharnos de esta función de Sage que devuelve el grupo de Galois de un polinomio y nuestros conocimientos de Teoría de Grupos, para escribir un programa que dado el grupo $\mathbb{Z}/n_1\mathbb{Z} \times \mathbb{Z}/n_2\mathbb{Z} \times \cdots \times \mathbb{Z}/n_l\mathbb{Z}$ como input[1] nos devuelva la lista $[n_1, n_2, \ldots, n_l]$. Una vez que sabemos la estructura del grupo ya podemos trabajar con el de forma más sencilla. Nos valemos primero de un programa que factorice un número y devuelva esta factorización en forma de lista. El programa por defecto de Sage hace más o menos eso, pero lo camufla para dar un aspecto más elegante. Si por ejemplo evaluamos dicho programa en el input 12 Sage nos devuelve $2^2 \cdot 3$. Nosotros queremos que el output sea $[4, 3]$, sencillamente porque es un objeto infórmatico más cómodo de manejar. Usaremos la estructura subyacente al programa por defecto de Sage para obtener nuestro programa en unas pocas y sencillas líneas.

[1] Obtenido mediante Sage como grupo de Galois de un polinomio.

```
def factorizacion(m):
   L=[];
   S = m.factor()
   for i in srange(len(S)):
      L.append((S[i][0])^(S[i][1]));
return L
```

También necesitaremos otro programa que ordene las listas de menor a mayor, para trabajar con más comodidad. Lo mostramos a continuación, así como un programa que busca el elemento máximo de una lista y que utilizaremos en el programa de ordenación. Ambos son programas sencillos.

```
def maximo(L):
   i=0;
   for j in srange(len(L)):
      if L[j]>i:
         i=L[j];
   return i

def reordenar(L):
   m = maximo(L);
   S=[];
   for j in [0..m]:
      if j in L:
         s = L.count(j);
         for i in srange(s):
            S.append(j);
   return S
```

Definimos ya el programa de comprobación mediante dos códigos, que se basan en el hecho de que Sage puede encontrar los generadores de un grupo abeliano finito G. Sabiendo eso, por Teoría de Grupos alcanzamos nuestro objetivo.

```
def pre_comprobacion(G):
   L=G.gens();
   S=[];
   for i in srange(len(L)):
      S.append(L[i].order())
   return S
```

```
def comprobacion(G):
   S = pre_comprobacion(G);
   J=[];
   for i in srange(len(S)):
      L = factorizacion(S[i]);
      for j in srange(len(L)):
         J.append(L[j]);
   J = reordenar(J);
   return J
```

El primer programa halla los órdenes de los generadores, mientras que el segundo halla la lista que buscamos dejandola en la forma más sencilla posible, esto es, ordenando de menor a mayor la lista y de tal manera que el mismo grupo no devuelva dos outputs distintos en el caso de que se pueda escribir de dos maneras distintas. Por ejemplo, como $\mathbb{Z}/12\mathbb{Z} \cong \mathbb{Z}/3\mathbb{Z} \times \mathbb{Z}/4\mathbb{Z}$, el programa en ambos casos devolverá $[3, 4]$. Así evitaremos más adelante confusiones generadas por posibles isomorfismos, ya que presentaremos los grupos siempre de esta forma.

Terminamos el apéndice con una explicación del programa definido en la sección 2.2 que resuelve el problema inverso de Galois para grupos abelianos finitos. Recordemos primero que aspecto tenía el programa para que la lectura sea más amena para el lector.

```
def GalInvProblemAbelianFiniteGroups(L):
   x = polygen(QQ,'x');
   if len(L)==1:
      return GalInvProblemCyclicGroup(L[0],1)
   L = reordenar(L);
   JJ=[];
   for i in srange(len(L)):
      if (L[i] in JJ)==False:
         JJ.append(L[i]);
   N=QQ;
   for i in srange(len(JJ)):
      s=1;
      l = L.count(JJ[i]);
      for j in srange(l):
         y = polygen(N,'y');
         G = GalInvProblemCyclicGroup(JJ[i],s);
         f = G[0].subs(x=y);
         while (f.is_irreducible())==False:
            s = s+1;
            G = GalInvProblemCyclicGroup(JJ[i],s);
            f = G[0].subs(x=y);
         M.<a> = NumberField(f);
         N.<b> = M.absolute_field();
   return N.defining_polynomial()
```

- Lo primero que hacemos es crear el anillo de polinomios $\mathbb{Q}[x]$ sobre el que generaremos los polinomios. A continuación reordenamos la lista `L`. Como vamos a usar la Proposición 2.2.1 para construir las extensiones sucesivamente, comenzamos construyendo las de orden menor.

- La lista `JJ` tiene como única función ver cuantos elementos distintos hay en la lista `L`. Por ejemplo, si `L` $= [2, 2, 2, 2, 3, 5]$, entonces `JJ`$= [2, 3, 5]$. Observar que `JJ` está ordenada por cuanto lo está `L`. El uso de `JJ` es fundamental para agilizar la ejecución, ya que nuestra idea es construir extensiones con grupo de Galois un grupo cíclico y mediante la Proposición 2.2.1 crear una extensión que tenga el producto de dichos grupos como grupo de Galois. Los grupos cíclicos del mismo orden necesitan polinomios distintos y por tanto necesitamos variar el segundo parámetro de nuestro programa de grupos cíclicos, tantas veces como el grupo aparezca en la lista `L`. Una vez que buscamos otro grupo cíclico podemos volver a bajar ese parámetro a 1 y así agilizar el programa.

- El resto del código programa la idea de la Proposición 2.2.1 de forma eficiente. Comenzamos trabajando sobre $\mathbb{Q}$ y miramos el primer elemento de la lista `JJ`. Después vemos cuantas veces aparece en `L` y creamos las correspondientes extensiones. Un dato importante es que los polinomios obtenidos están sobre $\mathbb{Q}$, por eso es importante en cada paso del bucle redefinir una variable y que convierta el polinomio en un polinomio en el anillo de polinomios de la extensión sobre la que trabajamos, para comprobar si es irreducible en dicha extensión o tenemos que buscar un polinomio nuevo. Una vez encontrado el polinomio, creamos la extensión `M` que es el cuerpo de descomposición de todos los polinomios obtenidos hasta el momento. Importante aquí es darse cuenta que Sage trata ésta `M` como una extensión sobre la anterior, por eso creamos `N` que consta de los mismos elementos pero esta vez se trata como extensión sobre $\mathbb{Q}$, que es lo que nos interesa.

APÉNDICE C

Complementos para la sección 4.3

En este Apéndice vamos a presentar los programas auxiliares necesarios para el correcto funcionamiento del programa presentado en la sección 4.3, que encuentra un polinomio con grupo de Galois A_n. Como el lector recordará, la base de estos programas está recogida en el Ejemplo 4.2.5, por lo que la implementación no es realmente complicada.

Vamos a programar primero el Teorema 3.1.2. El input es un polinomio y la acción del programa es sencilla: factorizamos el polinomio, comprobamos si todos los factores tienen multiplicidad 1 (en caso contrario rechazamos el polinomio porque la reducción no nos proporciona información alguna), y devolvemos una lista con los grados de cada factor irreducible que es lo que nos interesa. Una observación interesante es que aquí solo hacemos referencia a un polinomio, y no al anillo en el que se encuentra, y el Teorema 3.1.2 habla de polinomios en diferentes anillos. Podríamos especificar en el programa el anillo en el que trabajamos, pero en lugar de eso vamos a ejecutar el programa en los polinomios considerados dentro del anillo adecuado, en lugar de hacer reducciones dentro del programa.

```
def Dedekind(f):
   g = f.factor();
   for i in srange(len(g)):
      if g[i][1]!=1:
         return False
   L=[];
   for i in srange(len(g)):
      L.append(g[i][0].degree())
   return L
```

Una vez que tenemos descompuesto nuestro polinomio en un anillo de polinomios de un cuerpo finito y hemos obtenido los grados de cada factor, sabemos por el Teorema 3.1.2 que el grupo de Galois del polinomio posee una permutación de cierto tipo y nuestro interés es ver si elevando dicha permutación a una potencia natural podemos obtener un $p-$ciclo. En particular, debido al Lema 4.2.6, nos interesa encontrar un $3-$ciclo y un $p-$ciclo para cierto p. Es decir, nos interesa encontrar $p-$ciclos.

El siguiente programa distingue, dada una permutación de la forma $(a_1, a_2, \ldots, a_n)$, si una potencia de dicha permutación es un $p-$ciclo para un p primo. Como nuestra permutación viene dada en forma de lista por el programa `Dedekind`, no podemos hacer potencias de ella. Pero sí que podemos determinar como deben ser los números a_i para que exista alguna potencia en la que esta permutación se convierte en un $p-$ciclo.

En particular, vamos a utilizar que un $n-$ciclo elevado a un múltiplo de n es la identidad y que un $p-$ciclo elevado a un número que no sea múltiplo de p vuelve a ser un $p-$ciclo. Así que necesitamos exigirle tres cosas a los a_i:

1. Debe existir un $i \in \{1, \ldots, n\}$ tal que $a_i = p$. En efecto, de otro modo es imposible obtener un $p-$ciclo con potencias de nuestra permutación.
2. Para los i tales que $a_i \neq p$, debe tenerse que $p \nmid a_i$, así elevando la permutación al mínimo común múltiplo de los a_i obtenemos por cada $p-$ciclo presente en la permutación, un $p-$ciclo elevado a un número del que no es divisor, y por tanto volvemos a obtener un $p-$ciclo.
3. En relación a los dos puntos anteriores, podemos exigir que $a_i = p$ tan solo para un único i. De lo contrario al hacer la potencia descrita en el punto anterior obtendríamos un producto de $p-$ciclos, lo cual no siempre es un $p-$ciclo.

Así que el programa que nos interesa recibe como input un primo p y la lista obtenida en `Dedekind`, comprueba si verifica los tres requisitos y devuelve `True` en caso afirmativo y `False` en cualquier otro caso. Es decir, el programa discierne si de una permutación se puede obtener un $p-$ciclo o no.

```
def identificador_pciclos(p,L):
   for i in srange(len(L)):
      if (L[i]%p==0 and L[i]!=p) or (L.count(p)!=1):
         return False
   return True
```

Podemos ya escribir un programa que reciba el polinomio $g(y,t)$, haga reducciones a los primos hasta cierta tolerancia y compruebe, utilizando el Teorema 3.1.2, que el grupo de Galois del polinomio posee un $3-$ciclo y un $p-$ciclo, para un $p > \frac{n}{2}$. En esta ocasión mostramos el programa y después explicamos su funcionamiento.

Observación: Algunas lineas del programa, en particular algún condicional `if`, son demasiado largas como para estar contenidas en el espacio disponible, así que son cortadas en dos trozos para que quepan en la página. Para evitar que el lector confunda estos dos trozos con dos instrucciones distintas, se mantiene el sangrado del segundo trozo a la altura del primero, mientras que la siguiente línea tendrá una sangría más ya que será la consecuencia del condicional. Si el lector está interesado en implementar el código en Sage, debe abstenerse de realizar dicho corte y escribir cada instrucción en una línea, de lo contrario Sage no lo entenderá. Afortunadamente, Sage tiene espacio horizontal de sobra como para que no haya estos problemas de espacio.

```
def comprobador(n,coeficientes,primos):
   tresciclo = False;
   pciclo = False;
   for i in primos:
      y = polygen(Integers(i),'y');
      f = y^n+(coeficientes[1]%i)*y+(coeficientes[0]%i);
      if (Dedekind(f)!=False)
      and (identificador_pciclos(3,Dedekind(f))==True):
         tresciclo=True;
      for p in primos:
         if (p>n/2):
            if (Dedekind(f)!=False)
            and (identificador_pciclos(p,Dedekind(f))==True):
               pciclo=True;
      if (pciclo==True) and (tresciclo==True):
         return True
   return False
```

El input del programa no es nuestro polinomio en sí, si no tan solo el grado y los coeficientes de grado 0 y 1. La razón es que los polinomios que vamos a utilizar son siempre de una forma determinada[1] pero en distintos anillos, y de este modo podemos ir generando polinomios en cada anillo de una forma rápida. El otro input del programa es una lista en la cual vamos a incorporar todos los primos a los que nos interesa reducir el polinomio.

La idea del programa es bastante sencilla: partimos de dos variables `tresciclo`, `pciclo` a las que asignamos el valor `False` por defecto. Estas variables simbolizan la información que hemos obtenido en las reducciones que hemos realizado hasta el momento, en concreto, si en alguna reducción hemos demostrado ya que el grupo de Galois del polinomio posee un $3-$ciclo o un $p-$ciclo para $p > \frac{n}{2}$. Si en algún momento las dos variables registran `True`, entonces hemos hecho en nuestro polinomio las mismas comprobaciones que en el Ejemplo 4.2.5 a excepción de comprobar la irreducibilidad, este paso lo llevaremos a cabo en el programa principal.

En caso de que se agoten los primos de la lista que metemos como input y alguno de los valores siga siendo `False`, no podemos garantizar que se puedan llevar a cabo las comprobaciones del Ejemplo 4.2.5. No obstante, esto no quiere decir que nuestro polinomio no sea válido, puede ser que haciendo reducciones a primos mayores obtengamos lo que buscamos.

Ya hemos escrito todos los programas que se necesitan para el correcto funcionamiento del programa presentado en la sección 4.3, que mostramos de nuevo a continuación ya que el lector atento podrá ahora comprender su funcionamiento de un modo total.

[1] Recordamos que nos interesa la familia $g(y,t) = y^n - nty + (n-1)t$.

```
def ProbInvGalois_AlternatingGroup(n,ml,numerodeprimos):
   x = polygen(QQ,'x');
   primos = primes_first_n(numerodeprimos);
   while 1!=0:
      m=0;
      while m==0:
         m = randint(1,ml)/randint(1,ml);
      if n%4==0:
         a = n-1;
         b = n-1+m^2;
      elif n%4==1:
         a = m^2-n;
         b = m^2;
     elif n%4==2:
        a = n-1;
        b = n-1-m^2;
     else:
        a = n+m^2;
        b = m^2;
     t = a/b;
     g = x^n - n*t*x + (n-1)*t;
     if (g.is_irreducible)==True:
        if (t in ZZ)==True:
           if (comprobador(n,[(n-1)*t,-n*t],primos))==True:
              returng g
        if (t in ZZ)==False:
           if (comprobador(n,[(n-1)*a*(b^(n-1)),-n*a*(b^(n-2))],primos))
           ==True:
              return g
```

El programa funciona según toda nuestra discusión:

- Primero genera un $m \in \mathbb{Q}$ dentro de una tolerancia preconfigurada. Observar que aunque excluimos infinitas posibilidades de m esto no es importante del todo pues por la Proposición 4.3.2 a cada m le corresponde un t apropiado para nuestros intereses. No obstante el rango de búsqueda de m se puede variar a placer. No incluimos ningún número negativo en la búsqueda porque vamos a elevar m al cuadrado.

- Una vez construido m, el programa crea t según la Proposición 4.3.2. Observar que construimos numerador y denominador por separado, debido a que si $t \notin \mathbb{Z}$, necesitamos conocer esos valores para aplicar la Proposción 4.3.3.

- Una vez tenemos construido el polinomio, comprobamos que es irreducible y le aplicamos el programa `comprobador`, bien a nuestro polinomio, bien a su correspondiente polinomio en $\mathbb{Z}[x]$ obtenido por la Proposición 4.3.3. Notar que

no necesitamos verificar que el discriminante del polinomio es un cuadrado, ya que eso está garantizado por la Proposición 4.3.2.

- Como última observación, estamos utilizando un bucle infinito (`while 1!=0`). Este bucle simplemente se usa para asegurarnos de que encontramos un polinomio, ya que podríamos calcular un $t \in \mathbb{Q}$ que no fuera válido. En ese caso el programa no termina devolviendo el polinomio y por tanto entra de nuevo en el bucle.

APÉNDICE D

Método alternativo de resolución del problema para S_n

Como mencionamos al final de la sección 3.2, podemos escribir un programa que resuelva el problema inverso de Galois para S_n utilizando las mismas herramientas que en el caso de los grupos alternados y el hecho de que $g(y,t) = y^n - nty + (n-1)t$ tiene grupo de Galois S_n para infinitos racionales t. Análogamente a como hacíamos con los grupos alternados, vamos a llevar a cabo las comprobaciones del Ejemplo 4.2.2. Observar que en dicho ejemplo se comprueba que el discriminante del polinomio no es un cuadrado. Sin embargo esta comprobación no es necesaria y solo se realizaba para ilustrar que el grupo no era subgrupo de A_7.

Gracias al Lema 4.2.3 solo tenemos que modificar nuestro programa `comprobador` para que busque una transposición en lugar de un $3-$ciclo. Después repetiremos los pasos seguidos en el programa que resuelve el problema para grupos alternados escogiendo una especialización racional de $g(y,t)$.

```
def comprobador_2(n,coeficientes,primos):
   transposicion = False;
   pciclo = False;
   for i in primos:
      y = polygen(Integers(i),'y');
      f = y^n+(coeficientes[1]%i)*y+(coeficientes[0]%i);
      if (Dedekind(f)!=False)
      and (identificador_pciclos(2,Dedekind(f))==True):
         transposicion=True;
      for p in primos:
         if (p>n/2):
            if (Dedekind(f)!=False)
            and (identificador_pciclos(p,Dedekind(f))==True):
               pciclo=True;
      if (pciclo==True) and (transposicion==True):
         return True
   return False
```

```
def ProbInvGalois_SymmetricGroup2(n,ml,numerodeprimos):
   x = polygen(QQ,'x');
   primos = primes_first_n(numerodeprimos);
   while 1!=0:
      t=0;
      while t==0:
         a = randint(1,ml);
         b = randint(1,ml);
         t=a/b;
      g = x^n-n*t*x + (n-1)*t;
      if (g.is_irreducible())==True:
         if (t in ZZ)==True:
            if (comprobador_2(n,[(n-1)*t,-n*t],primos))==True:
               return g
         if (m in ZZ)==False:
            if (comprobador_2(n,[(n-1)*a*(b^(n-1)),-n*a*(b^(n-2))],primos))
            ==True:
               return g
```

Observar que aunque este programa sea más eficiente que el propuesto en la sección 3.2, siempre devuelve polinomios de la forma $x^n - ntx + (n-1)t$. Los polinomios mostrados en el apéndice F se han calculado usando el programa de la sección 3.2.

APÉNDICE E

Polinomios para algunos grupos abelianos finitos

Grupo	Polinomio
$\mathbb{Z}/2\mathbb{Z}$	$x^2 + x + 1$
$\mathbb{Z}/3\mathbb{Z}$	$x^3 + x^2 - 2x - 1$
$\mathbb{Z}/4\mathbb{Z}$	$x^4 + x^3 + x^2 + x + 1$
$\mathbb{Z}/2\mathbb{Z} \times \mathbb{Z}/2\mathbb{Z}$	$x^4 + x^2 + 9$
$\mathbb{Z}/5\mathbb{Z}$	$x^5 + x^4 - 4x^3 - 3x^2 + 3x + 1$
$\mathbb{Z}/6\mathbb{Z}$	$x^6 + x^5 + x^4 + x^3 + x^2 + x + 1$
$\mathbb{Z}/7\mathbb{Z}$	$x^7 + x^6 - 12x^5 - 7x^4 + 28x^3 + 14x^2 - 9x + 1$
$\mathbb{Z}/8\mathbb{Z}$	$x^8 + x^7 - 7x^6 - 6x^5 + 15x^4 + 10x^3 - 10x^2 - 4x + 1$
$\mathbb{Z}/2\mathbb{Z} \times \mathbb{Z}/4\mathbb{Z}$	$x^8 - 2x^7 - 8x^5 + 27x^4 - 62x^3 + 117x^2 - 23x + 29$
$\mathbb{Z}/2\mathbb{Z} \times \mathbb{Z}/2\mathbb{Z} \times \mathbb{Z}/2\mathbb{Z}$	$x^8 + 4x^7 + 20x^6 + 46x^5 + 123x^4 + 174x^3 - 88x^2 - 168x + 144$
$\mathbb{Z}/9\mathbb{Z}$	$x^9 + x^8 - 8x^7 - 7x^6 + 21x^5 + 15x^4 - 20x^3 - 10x^2 + 5x + 1$
$\mathbb{Z}/3\mathbb{Z} \times \mathbb{Z}/3\mathbb{Z}$	$x^9 - 20x^7 + 8x^6 + 103x^5 - 46x^4 - 157x^3 + 32x^2 + 60x - 8$
$\mathbb{Z}/10\mathbb{Z}$	$x^{10} + x^9 + x^8 + x^7 + x^6 + x^5 + x^4 + x^3 + x^2 + x + 1$
$\mathbb{Z}/11\mathbb{Z}$	$x^{11} + x^{10} - 10x^9 - 9x^8 + 36x^7 + 28x^6 - 56x^5 - 35x^4 + 35x^3 + 15x^2$ $-6x - 1$
$\mathbb{Z}/12\mathbb{Z}$	$x^{12} + x^{11} + x^{10} + x^9 + x^8 + x^7 + x^6 + x^5 + x^4$ $+x^3 + x^2 + x + 1$
$\mathbb{Z}/2\mathbb{Z} \times \mathbb{Z}/6\mathbb{Z}$	$x^{12} - 4x^{11} - 11x^{10} + 52x^9 + 25x^8 - 206x^7 + 48x^6 + 257x^5 - 129x^4$ $-65x^3 + 28x^2 + 6x - 1$
$\mathbb{Z}/13\mathbb{Z}$	$x^{13} + x^{12} - 24x^{11} - 19x^{10} + 190x^9 + 116x^8 - 601x^7 - 246x^6$ $+738x^5 + 215x^4 - 291x^3 - 68x^2 + 10x + 1$
$\mathbb{Z}/14\mathbb{Z}$	$x^{14} + x^{13} - 13x^{12} - 12x^{11} + 66x^{10} + 55x^9 - 165x^8$ $-120x^7 + 210x^6 + 126x^5 - 126x^4 - 56x^3 + 28x^2 + 7x - 1$
$\mathbb{Z}/15\mathbb{Z}$	$x^{15} + x^{14} - 14x^{13} - 13x^{12} + 78x^{11} + 66x^{10} - 220x^9$ $-165x^8 + 330x^7 + 210x^6 - 252x^5 - 126x^4 + 84x^3 + 28x^2 - 8x - 1$

APÉNDICE F

Polinomios para algunos grupos simétricos y alternados

Grupo	Polinomio
S_3	$x^3 - 2$
S_4	$x^4 + 24x^3 + 32x^2 - 5x - 15$
S_5	$x^5 + 44x^4 - 15x^2 + 38x - 9$
S_6	$x^6 + 24x^5 - 15x^4 - 15x^3 + 38x^2 + x - 15$
S_7	$x^7 + 24x^6 + 32x^5 + 34x^3 + 44x^2 + 5x - 9$
S_8	$x^8 + 24x^7 + 12x^6 + 9x^4 + 29x^3 - 15x^2 + 16x - 15$
S_9	$x^9 + 24x^8 + 12x^7 + 20x^6 + 10x^5 - 15x^4 + 38x^3 + 20x^2 + 38x - 9$
S_{10}	$x^{10} + 24x^9 + 12x^8 - 15x^6 - 15x^5 + 38x^4 + 5x^3 + 13x^2 + x - 15$

Grupo	Polinomio
A_3	$x^3 - \frac{21}{4}x + \frac{7}{2}$
A_4	$x^4 - 3x + \frac{9}{4}$
A_5	$x^5 + 20x - 16$
A_6	$x^6 - 6x + 5$
A_7	$x^7 - 56x + 48$
A_8	$x^8 - 7x + \frac{49}{8}$
A_9	$x^9 + 72x - 64$
A_{10}	$x^{10} - \frac{45}{4}x + \frac{81}{8}$
A_{11}	$x^{11} - 132x + 120$
A_{12}	$x^{12} - 11x + \frac{121}{12}$
A_{13}	$x^{13} - \frac{156}{25}x + \frac{144}{25}$
A_{14}	$x^{14} - \frac{91}{6}x + \frac{169}{12}$
A_{15}	$x^{15} - 240x + 224$

Bibliografía

[1] Conrad, K., *Recognizing Galois Groups S_n and A_n*. Disponible en https://kconrad.math.uconn.edu/blurbs/galoistheory/galoisSnAn.pdf

[2] Conrad, K., *Galois Groups as Permutation Groups*. Disponible en https://kconrad.math.uconn.edu/blurbs/galoistheory/galoisaspermgp.pdf

[3] Cox, D., *Galois Theory*. John Wiley & Sons, Inc. Hoboken, New Jersey, 2012.

[4] Fraleigh, J.B. , *Álgebra Abstracta*. Addison-Wesley Iberoamericana, Wilmington, Delaware, 1988.

[5] Jensen, C.U. Leder, A. Yui, N., *Generic Polynomials. Constructive Aspects of the Galois Inverse Problem*. Cambridge University Press, USA, 2002.

[6] Kumar, M. Shekar,G. Misra, L., *A study on the Inverse Galois Problem in Galois Theory*. International Journal of modern Electronics and Communication Engineering, Volume No.-3, Issue No.-3, September 2015.

[7] The Sage Developers, *SageMath, The Sage Mathematics Software System Version 8.4*. Disponible en https://www.sagemath.org

[8] van der Waerden, B.L., *Modern Algebra*. Springer-Verlag, 1930.

[9] Viola-Prioli, A.M. Viola-Prioli, J.E., *Teoría de Cuerpos y Teoría de Galois*. Editorial Reverté, S.A , Barcelona, España, 2006.