7510 HW 2

Duncan Wilkie

28 August 2022

1.6.18. *For any group G, the map from G to itself defined by $f : g \mapsto g^2$ is a homomorphism iff G is Abelian.*

Proof. If G is Abelian, $f(gh) = (gh)^2 = g^2h^2 = f(g)f(h)$. Conversely, if $f : g \mapsto g^2$ is a homomorphism, $f(hg) = (hg)^2 = hghg$. By the homomorphism property, $f(hg) = h^2g^2$, so

$$h^2g^2 = hghg \Leftrightarrow hg = gh$$

so G must be Abelian. □

1.6.23. *Let G be a finite group which possesses an automorphism σ such that $\sigma(g) = g$ iff $g = 1$. If σ^2 is the identity map from G to G, prove that G is Abelian.*

Proof. Let $G \ni x \overset{f}{\mapsto} x^{-1}\sigma(x) \in G$. We prove that f is injective (and therefore bijective, since G is finite), so every element of G can be written $g = x^{-1}\sigma(x) = f^{-1}(g)$. If $f(x) = f(y)$, then

$$x^{-1}\sigma(x) = y^{-1}\sigma(y) \Leftrightarrow yx^{-1} = \sigma(y)\sigma(x)^{-1} = \sigma(y)\sigma(x^{-1}) = \sigma(yx^{-1})$$

Since σ is fixed-point-free, this implies $yx^{-1} = 1 \Leftrightarrow y = x$.

We may therefore write

$$\sigma(g) = \sigma(x^{-1}\sigma(x)) = \sigma(x)^{-1}x = (x^{-1}\sigma(x))^{-1} = g^{-1}$$

Then $\sigma : g \mapsto g^{-1}$; in particular, since it is also a homomorphism,

$$gh = (h^{-1}g^{-1})^{-1} = \sigma(h^{-1}g^{-1}) = \sigma(h^{-1})\sigma(g^{-1}) = hg.$$

□

1.7.18. *If $H \circlearrowright A$, show that the relation $\sim$ on A defined by*

$$a \sim b \Leftrightarrow \exists h \in H : a = hb$$

is an equivalence relation.

Proof. It's reflexive: $a = ha$ must hold for $h = 1$ (which must be an element of H) by the corresponding action axiom. It's also symmetric: if $a = hb$, then

$$b = 1 \cdot b = h^{-1}(hb) = h^{-1}a$$

Transitivity follows from associativity. If $a \sim b$ and $b \sim c$,

$$a = h_1b = h_1(h_2c) = (h_1h_2)c,$$

so $a \sim c$. □

2.1.5. *A group G with $n = |G| > 2$ cannot have a subgroup H with $|H| = n - 1$.*

Proof. Suppose such a subgroup exists. Then there exists exactly one element $g \in G - H$. $n > 2$ means $|H| \geq 2$, i.e. there is a nonidentity element $h \in H$. Take gh in G; since h is nonidentity, $gh = h' \neq g$, as otherwise $gh = g \Leftrightarrow h = 1$ (so $h' \in H$). But the equivalent formula $g = h'h^{-1}$ would mean by closure of H under inverses and products that $g \in H$, a contradiction. □

2.1.6. *If G is an Abelian group, then $G_T = \{g \in G \mid |G| < \infty\} \leq G$. Give a counterexample when G is not Abelian.*

Proof. Take arbitrary $x, y \in G_T$. Let $n = |x|$, $m = |y|$, and $l = \operatorname{lcm}(n, m) = an = bm$. Since G is Abelian, we can distribute exponents, allowing us to write

$$(xy^{-1})^l = x^l y^{-l} = (x^n)^a (y^m)^{-b} = 1^a 1^{-b} = 1$$

which shows that $xy^{-1} \in G_T$ (its order is at most l); consequently, $G_T \leq G$.

Consider the matrices in $GL_2(\mathbb{R})$.

$$\begin{pmatrix} 1 & 1 \\ 0 & -1 \end{pmatrix}^2 = \begin{pmatrix} 1 & 0 \\ 1 & -1 \end{pmatrix}^2 = I$$

However,

$$\begin{pmatrix} 1 & 1 \\ 0 & -1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 1 & -1 \end{pmatrix} = \begin{pmatrix} 2 & -1 \\ -1 & 2 \end{pmatrix}$$

has that

$$\begin{pmatrix} 2 & -1 \\ -1 & 2 \end{pmatrix} \begin{pmatrix} 2 & -1 \\ -1 & 2 \end{pmatrix} = \begin{pmatrix} 2^2 & (-1)^2 \\ (-1)^2 & 2^2 \end{pmatrix}$$

This provides an inductive base case; presuming the form in which I've chosen to write the matrix to generalize to the nth power,

$$\begin{pmatrix} 2 & -1 \\ -1 & 2 \end{pmatrix}^{n+1} = \begin{pmatrix} 2 & -1 \\ -1 & 2 \end{pmatrix}^n \begin{pmatrix} 2 & -1 \\ -1 & 2 \end{pmatrix} = \begin{pmatrix} 2^n & (-1)^n \\ (-1)^n & 2^n \end{pmatrix} \begin{pmatrix} 2 & -1 \\ -1 & 2 \end{pmatrix} = \begin{pmatrix} 2^{n+1} & (-1)^{n+1} \\ (-1)^{n+1} & 2^{n+1} \end{pmatrix}$$

Since for no $n \geq 1$ is any entry equal to the corresponding entry in the identity matrix, the order of the product is infinite. Therefore, $GL_2(\mathbb{R})_T \not\leq GL_2(\mathbb{R})$, as we've demonstrated two elements of the former whose product isn't also in it. □

2.4.9. *Prove that $SL_2(\mathbb{F}_3)$ is the subgroup of $GL_2(\mathbb{F}_3)$ generated by $r = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}$ and $s = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}$*

Proof. Denote the generated group by G. Note that $G \leq SL_2(\mathbb{F}_3)$: the generators have determinant 1, so the generators are indeed a subset of $SL_2(\mathbb{F}_3)$, and it's immediately a subgroup like all generated subgroups.

We now prove the reverse relation, that $SL_2(\mathbb{F}_3) \leq G$. Since we don't have the tools necessary to do this simply, there are few better ways than an exhaustive enumeration. It feels more mathematically sophisticated to write an algorithm from scratch to do so, rather than writing it out by hand. The following Guile Scheme program implements a procedure (that extends trivially to $\mathbb{F}_n$ and to different generators, and easily to SL_n) for expressing every element of $SL_2(\mathbb{F}_3)$ as a product of the given generators. The comments hopefully indicate the steps comprehensibly.

```
(use-modules (srfi srfi-1) (ice-9 pretty-print))

;; Generator candidates
(define r '((1 1) (0 1)))
(define s '((1 0) (1 1)))

;; Shorthand for the modular arithmetic operations.
;; Replace these with some $n \neq 3$ to extend to $\mathbb{F}_3$
(define (+mod3 x y) (modulo (+ x y) 3))
(define (*mod3 x y) (modulo (* x y) 3))

;; Calculate the determinant of a 2-by-2 matrix.
;; Replace with the more complicated $n$-by-$n$ version to extend beyond $GL_2$.
(define (det matrix)
  (modulo (- (* (caar matrix) (cadadr matrix))
             (* (cadar matrix) (caadr matrix)))
          3))

;; Multiply two matrices (general)
(define (mmult mat1 mat2)
  (map
   (lambda (row)
     (apply map
            (lambda column
              (apply +mod3 (map *mod3 row column)))
            mat2))
   mat1))



;; Take the Cartesian product of any number of lists.
(define (cart-product lists)
  (fold-right (lambda (xs ys)
                (append-map (lambda (x)
                              (map (lambda (y)
                                     (cons x y))
                                   ys))
                            xs))
              '(())
              lists))


;; Use the Cartesian product to take the nth Cartesian power of a particular list.
;; The words of length n formed by a list of generators are precisely
;; the $n$th Cartesian power of the list of generators (ordered selection with replacement).

```

```
(define (cart-power xs n)
  (if (= n 1)
      (map list xs)
      (cart-product (map (lambda (-) xs)
                         (iota n)))))

;; Utility function that divides a list into equal parts,
;; e.g. (1 2 3 4) → ((1 2) (3 4)) or (1 2 3 4 5 6) → ((1 2 3) (4 5 6))
(define (chunk xs n)
  (if (null? xs)
      '()
      (let ((head (take xs n))
            (rest (drop xs n)))
        (cons head (chunk rest n)))))

;; Generate all elements of $SL_2(\mathbb{F}_3)$ by filtering
;; out 2-by-2 matrices with entries in {0, 1, 2} without unit determinant.
(define sl2mod3
  (filter (lambda (mat) (= (det mat) 1))
          (map (lambda (l) (chunk l 2))
               (cart-power '(0 1 2) 4))))


;; Recursively evaluate a word.

;; 1-words evaluate to themselves.
;; If the word's length is even, associate pairs, multiply the pairs,
;; and evaluate the resulting word of length $\frac{1}{2}n$.
;; If it's odd, multiply the first element of the word with the result of evaluating the rest.
(define (eval-word word)
  (if (= (length word) 1)
      (car word)
      (if (= (modulo (length word) 2) 0)
          (eval-word
           (map mmult (map car (chunk word 2)) (map cadr (chunk word 2))))
          (mmult (car word) (eval-word (cdr word))))))




;; Given an element and a set of generators (and a cutoff length, to handle non-termination),
;; search for a product of the generators equalling the element.

;; Starting from length-1 words, create a list of all words consisting of generators.
;; Search through that list to find and return any words that evaluate to the given element;
;; failing that, increment the length and repeat.
```

```
(define (write-as-word elt gens maxlen)
  (let words ((len 1))
    (if (> len maxlen)
        #f
        (let ((match (find (lambda (word) (equal? (eval-word word) elt))
                           (cart-power gens len))))
          (if match
              match
              (words (1+ len)))))))


;; Generate a pairing of elements of SL_2(F_3) with the words of r and s that equal them.
(define presentation (zip sl2mod3 (map (lambda (sl2)
                                     (write-as-word sl2 (list r s) 5)) sl2mod3)))

;; Display the result of the above in a way easier on the eyes.
(do ((i 0 (1+ i)))
    ((>= i (length presentation) 1))
  (pretty-print "-----------------------\n" #:display? #t)
  (pretty-print (car (list-ref presentation i)) #:max-expr-width 6)
  (pretty-print "\n→\n" #:display? #t)
  (do ((j 0 (1+ j)))
      ((>= j (length (cadr (list-ref presentation i))) 1))
    (pretty-print (list-ref (cadr (list-ref presentation i)) j) #:max-expr-width 6)
    (pretty-print "\n" #:display? #t)))
```

This indeed produces a representation for all 24 elements of $SL_2(\mathbb{F}_3)$, but the output is rather long for my printing supplies budget. Copying the code verbatim into the GNU Guile REPL should reproduce it. $\square$

2.4.14. *A group H is called finitely generated if $H = \langle A \rangle$ for some finite set A. Every finite group is finitely generated, $\mathbb{Z}$ is finitely generated, and every finitely generated subgroup of the additive group $\mathbb{Q}$ is cyclic.*

Proof. For a finite group, take $A = H$. The group so-generated is exactly H from the intersection definition, since H is the only subgroup such that $H \subseteq H$. For $\mathbb{Z}$, take $A = \{1\}$. Since $\mathbb{Z}$ is Abelian, we can write $\langle 1 \rangle = \{n \cdot 1 \mid n \in \mathbb{Z}\} = \mathbb{Z}$, so $\langle A \rangle = \mathbb{Z}$. A finitely generated subgroup of $\mathbb{Q}$ has generators $\frac{p_1}{q_1}, \frac{p_2}{q_2}, \cdots \frac{p_n}{q_n}$ for $q_i, p_i \in \mathbb{Z}$. Write $k = \prod_{1 \le i \le n} q_i$; every generator can be written as a repeated sum (possibly empty or negative times, since $1, \frac{-1}{k} \in \langle 1 \rangle$) of $\frac{1}{k}$, since

$$\sum_{p_i k/q_i \text{ times}} \frac{1}{k} = \frac{p_i k}{q_i} \frac{1}{k} = \frac{p_i}{q_i}$$

This further implies that every element of the generated subgroup is a repeated sum of $\frac{1}{k}$, since such elements are sums of generators and their inverses, i.e. $\langle A \rangle \le \langle \frac{1}{k} \rangle$. However $\langle \frac{1}{k} \rangle$ is cyclic, and every subgroup of a cyclic group is cyclic. $\square$

Proposition 1. *Let F be a field and X be the set of 1-dimensional vector subspaces of F^n. Let U be the group of upper-triangular $n \times n$ matrices with 1's on the diagonal. Then U acts on X. Determine the orbits of the action.*

Proof. The orbit of a particular element V of X is the set of all other elements of X that can be reached by acting on the particular element. In Einstein notation, a fixed $f \in V \in X$ is a vector in F^n of the form $f = (0, \cdots, 0, f_i, 0, \cdots 0) \Rightarrow f_k = \delta_{ki} f_i$. When $u \in U$ acts on elements f by matrix multiplication, the output is $(uf)_j = u_{jk} f_k = u_{jk}(\delta_{ki} f_i) = u_{ji} f_i$, which is f_i times the ith column vector of u. These vectors have arbitrary elements $f_i u_{ji}$ for $1 \leq j < i$, are equal to f_i for $j = i$, and are zero for $i < j \leq n$. Any particular one-dimensional subspace is the collection of vectors with all possible values of f_i with fixed i, so each $u \in U$ has an orbit on each 1D input subspace of the 1D subspace consisting of constant multiples of the ith column vector of u.

□