Chess Notes

James Otto

11/15/22

Table of contents

Preface

These are my notes on various topics in chess, aggregated from a variety of sources.

This is a Quarto book. To learn more about Quarto books visit https://quarto.org/docs/books.

Part I

Openings

1 The Scotch

```
board_setup = function(data, title) {

  var game = data.find(d => d.title === title)

  // if there's just one variation, need to put in array
  var variations = game.variations

  var pos = game.start - 1
  var moved = false

  var onDragMove = function() {
    moved = true

    d3.select('#reset-' + title)
      .classed('btn-outline-secondary', false)
      .classed('btn-success', true)

    d3.select('#analyze-' + title).property("disabled", true)
    d3.select('#prev-' + title).property("disabled", true)
    d3.select('#next-' + title).property("disabled", true)
  }

  // Note: boolean arrays don't survive transpose(), need to use characters ("show" and "h
  var board = Chessboard(title, {showNotation: game.notation === 'show', position: game.po

  // resize board if window is resized
  $(window).resize(board.resize)

  d3.select('#prev-' + title).property("disabled", pos == 0)
  d3.select('#next-' + title).property("disabled", pos == game.positions.length - 1)

  $('#next-' + title).on('click', function () {
    if (pos < game.positions.length - 1) {
      pos = pos + 1
```

```
    board.position(game.positions[pos])
  }

  d3.select('#analyze-' + title).property("disabled", false)
  d3.select('#prev-' + title).property("disabled", false)
  d3.select('#next-' + title).property("disabled", pos == game.positions.length - 1)
})

$('#prev-' + title).on('click', function () {
  if (pos > 0) {
    pos = pos - 1
    board.position(game.positions[pos])
  }

  d3.select('#analyze-' + title).property("disabled", false)
  d3.select('#prev-' + title).property("disabled", pos == 0)
  d3.select('#next-' + title).property("disabled", false)
})

$('#reset-' + title).on('click', function() {

  // if pieces have not been moved, reset to beginning of line
  if (!moved) {

    // if looking at a variation, reset to main line
    // (otherwise, just reset current line)
    if (!(variations === undefined) && variations.includes(game.title)) {

      d3.select('#' + game.title)
        .style("box-shadow", null)
        .style("color", null)

      game = data.find(d => d.title === title)

      d3.select('#reset-' + title)
        .classed('btn-primary', false)
        .classed('btn-outline-secondary', true)

    }

    // if reset button is pressed at starting position, totally reset board
```

```
    if (pos === game.start - 1) {
      pos = 0
    } else {
      pos = game.start - 1
    }

  } else {

    // color reset button correctly if pieces are moved:
    if (!(variations === undefined) && variations.includes(game.title)) {

      d3.select('#reset-' + title)
        .classed('btn-primary', true)
        .classed('btn-success', false)

    } else {

      d3.select('#reset-' + title)
        .classed('btn-outline-secondary', true)
        .classed('btn-success', false)

    }
  }

  board.position(game.positions[pos])

  d3.select('#analyze-' + title).property("disabled", false)
  d3.select('#prev-' + title).property("disabled", pos == 0)
  d3.select('#next-' + title).property("disabled", pos == game.positions.length - 1)

  moved = false

})

$('#analyze-' + title).on('click', function() {
  window.open(game.link[pos], '_blank');
})

$('#flip-' + title).on('click', board.flip)
```

```
if (!(variations === undefined)) {

  if (!(Array.isArray(variations))) {
    variations = [variations]
  }

  for (let k = 0; k < variations.length; k++) {
    $('#' + variations[k]).on('click', function() {

      // if clicked on active variation, reset to main line
      if (game.title === variations[k]) {

        d3.select('#' + game.title)
          .style("box-shadow", null)
          .style("color", null)

        game = data.find(d => d.title === title)

        // Don't reset position if before start of game
        if (pos >= game.start) {
          pos = game.start - 1
        }

        board.position(game.positions[pos])

        d3.select('#reset-' + title)
          .classed('btn-primary', false)
          .classed('btn-outline-secondary', true)

      // otherwise, set to selected variation
      } else {

        game = data.find(d => d.title === variations[k])

        // Don't reset position if before start
        // (this allows for the complete viewing of variations)
        if (pos >= game.start) {
          pos = game.start - 1
        }

        board.position(game.positions[pos])
```

```
          d3.select('#reset-' + title)
            .classed('btn-outline-secondary', false)
            .classed('btn-success', false)
            .classed('btn-primary', true)

        // need to unset styling for other variations
        variations.filter(name => !(name === variations[k]))
          .map(name =>  {
            d3.select('#' + name)
              .style("box-shadow", null)
              .style("color", null)
              //.style("box-shadow", "inset 0 0 0 0 #2780e3")
              //.style("color", "#2780e3")
          })

        // indicate variation is selected
        d3.select('#' + variations[k])
          .style("box-shadow", "inset 10vw 0 0 0 #2780e3")
          .style("color", "white")

        }

        // activate/deactivate UI
        d3.select('#analyze-' + title).property("disabled", false)
        d3.select('#prev-' + title).property("disabled", pos == 0)
        d3.select('#next-' + title).property("disabled", pos == game.positions.length - 1)

      })
    }

  }

}
```

```
scotch = transpose(scotch_t)
```

```
scotch.filter(d => d.type === 'main')
  .map(d => board_setup(scotch, d.title))
```

1.1 Introduction

1.2 Main Line (4.Nxd4)

Classical Variation (4...Bc5)

- After 4.Nxd4 Bc5 there are three main lines:
  - 5.Nxc6
  - 5.Be3
  - 5.Nb3

Schmidt Variation (4...Nf6)

- After 4.Nxd4 Nf6 there are two options:
  - The Mieses Variation: 5.Nxc6
  - The Scotch Four Knights Game: 5.Nc3

Steinitz Variation (4...Qh4!?)

- Black wins a pawn here but gives white a lead in development and ends up with their king in a slightly awkward spot.

1.3 Scotch Gambit (4.Bc4)

- Black can accept with 4...Bc5 5.c3 dxc3 .
- Alternatively, 5...Nf6 transitions into the Italian Game
- 4...Bb4+ is the London Defense

1.4 Traps

Classical Variation (4.Nxd4 Bc5)

- If black doesn't defend the bishop on c5 after 4.Nxd5 Bc5 5.Be3, white can take the knight on c6, winning the bishop!
  - For example, 5....Nf6 loses the bishop
  - 5....d6 defends the bishop, but it's not pretty

todo: - 3…d6 - Göring Gambit

2 Fried Liver Attack

```
board_setup = function(data, title) {

  var game = data.find(d => d.title === title)

  // if there's just one variation, need to put in array
  var variations = game.variations

  var pos = game.start - 1
  var moved = false

  var onDragMove = function() {
    moved = true

    d3.select('#reset-' + title)
      .classed('btn-outline-secondary', false)
      .classed('btn-success', true)

    d3.select('#analyze-' + title).property("disabled", true)
    d3.select('#prev-' + title).property("disabled", true)
    d3.select('#next-' + title).property("disabled", true)
  }

  // Note: boolean arrays don't survive transpose(), need to use characters ("show" and "h
  var board = Chessboard(title, {showNotation: game.notation === 'show', position: game.po

  // resize board if window is resized
  $(window).resize(board.resize)

  d3.select('#prev-' + title).property("disabled", pos == 0)
  d3.select('#next-' + title).property("disabled", pos == game.positions.length - 1)

  $('#next-' + title).on('click', function () {
    if (pos < game.positions.length - 1) {
      pos = pos + 1
```

```
    board.position(game.positions[pos])
  }

  d3.select('#analyze-' + title).property("disabled", false)
  d3.select('#prev-' + title).property("disabled", false)
  d3.select('#next-' + title).property("disabled", pos == game.positions.length - 1)
})

$('#prev-' + title).on('click', function () {
  if (pos > 0) {
    pos = pos - 1
    board.position(game.positions[pos])
  }

  d3.select('#analyze-' + title).property("disabled", false)
  d3.select('#prev-' + title).property("disabled", pos == 0)
  d3.select('#next-' + title).property("disabled", false)
})

$('#reset-' + title).on('click', function() {

  // if pieces have not been moved, reset to beginning of line
  if (!moved) {

    // if looking at a variation, reset to main line
    // (otherwise, just reset current line)
    if (!(variations === undefined) && variations.includes(game.title)) {

      d3.select('#' + game.title)
        .style("box-shadow", null)
        .style("color", null)

      game = data.find(d => d.title === title)

      d3.select('#reset-' + title)
        .classed('btn-primary', false)
        .classed('btn-outline-secondary', true)

    }

    // if reset button is pressed at starting position, totally reset board
```

```
    if (pos === game.start - 1) {
      pos = 0
    } else {
      pos = game.start - 1
    }

  } else {

    // color reset button correctly if pieces are moved:
    if (!(variations === undefined) && variations.includes(game.title)) {

      d3.select('#reset-' + title)
        .classed('btn-primary', true)
        .classed('btn-success', false)

    } else {

      d3.select('#reset-' + title)
        .classed('btn-outline-secondary', true)
        .classed('btn-success', false)

    }
  }

  board.position(game.positions[pos])

  d3.select('#analyze-' + title).property("disabled", false)
  d3.select('#prev-' + title).property("disabled", pos == 0)
  d3.select('#next-' + title).property("disabled", pos == game.positions.length - 1)

  moved = false

})

$('#analyze-' + title).on('click', function() {
  window.open(game.link[pos], '_blank');
})

$('#flip-' + title).on('click', board.flip)
```

```
if (!(variations === undefined)) {

  if (!(Array.isArray(variations))) {
    variations = [variations]
  }

  for (let k = 0; k < variations.length; k++) {
    $('#' + variations[k]).on('click', function() {

      // if clicked on active variation, reset to main line
      if (game.title === variations[k]) {

        d3.select('#' + game.title)
          .style("box-shadow", null)
          .style("color", null)

        game = data.find(d => d.title === title)

        // Don't reset position if before start of game
        if (pos >= game.start) {
          pos = game.start - 1
        }

        board.position(game.positions[pos])

        d3.select('#reset-' + title)
          .classed('btn-primary', false)
          .classed('btn-outline-secondary', true)

      // otherwise, set to selected variation
      } else {

        game = data.find(d => d.title === variations[k])

        // Don't reset position if before start
        // (this allows for the complete viewing of variations)
        if (pos >= game.start) {
          pos = game.start - 1
        }

        board.position(game.positions[pos])
```

```
          d3.select('#reset-' + title)
            .classed('btn-outline-secondary', false)
            .classed('btn-success', false)
            .classed('btn-primary', true)

        // need to unset styling for other variations
        variations.filter(name => !(name === variations[k]))
          .map(name =>  {
            d3.select('#' + name)
              .style("box-shadow", null)
              .style("color", null)
              //.style("box-shadow", "inset 0 0 0 0 #2780e3")
              //.style("color", "#2780e3")
          })

        // indicate variation is selected
        d3.select('#' + variations[k])
          .style("box-shadow", "inset 10vw 0 0 0 #2780e3")
          .style("color", "white")

        }

        // activate/deactivate UI
        d3.select('#analyze-' + title).property("disabled", false)
        d3.select('#prev-' + title).property("disabled", pos == 0)
        d3.select('#next-' + title).property("disabled", pos == game.positions.length - 1)

      })
    }

  }

}
```

```
fried_liver = transpose(fried_liver_t)
```

```
fried_liver.map(d => board_setup(fried_liver, d.title))
```

2.1 Introduction

2.2 Continuing

References