VE1ATM HF Propagation Prediction System

A web-based HF propagation prediction tool customized for VE1ATM (FN74ui) with DXCC tracking integration.

Features

✅ **Real-time solar-terrestrial data** (SFI, SSN, Kp, A-index)
✅ **Band-by-band predictions** for 40m through 10m
✅ **Regional propagation quality** (EU, JA, VK, ZL, etc.)
✅ **24-hour forecast timeline**
✅ **DXCC progress tracking** with most-wanted list
✅ **Interactive world map** showing current openings
✅ **Beautiful dark-mode UI** optimized for readability

Files

- `generate_propagation.py` - Python script to generate predictions
- `propagation_dashboard.html` - Web interface
- `propagation_data.json` - Generated prediction data (auto-created)
- `dxcc_summary.json` - Your DXCC worked/needed entities (from log)

Usage

1. Generate Fresh Predictions

Run the Python script whenever you want updated predictions:

bash

```
python3 generate_propagation.py
```

This will:

- Fetch current solar data from NOAA (or use typical values if offline)
- Calculate propagation for all bands to major regions
- Generate 24-hour forecast
- Save results to `propagation_data.json`

2. View the Dashboard

Simply open `propagation_dashboard.html` in any web browser. The page will automatically load the latest predictions.

Note: For the map and styling to work properly, you need to either:

- Open from a web server (even `python3 -m http.server`)

- Or allow local file access in your browser

3. Update Schedule

For best results, regenerate predictions:

- **Every 1-3 hours** for current conditions
- **When solar conditions change** (after solar flares, CMEs, etc.)
- **Before operating sessions** to plan band strategy

You can automate this with a cron job:

bash

```
# Run every 2 hours
0 */2 * * * cd /path/to/predictions && python3 generate_propagation.py
```

Understanding the Display

Solar Conditions Panel

- **SFI (Solar Flux Index)**: Higher = better HF propagation (typical: 70-250)
- **Sunspot Number**: Indicates solar cycle position (0-300+)
- **Kp Index**: Geomagnetic activity (0-9, lower is better)
- **A Index**: Geomagnetic field stability (0-400, lower is better)

Band Status Cards

- **Green (GOOD)**: Reliability > 60%, SNR > 10dB - Excellent conditions
- **Yellow (FAIR)**: Reliability 30-60%, SNR 0-10dB - Marginal conditions
- **Red (POOR)**: Reliability < 30%, SNR < 0dB - Poor conditions
- **Gray (CLOSED)**: Above MUF - Band closed to that region

Quality Metrics

- **Reliability %**: Probability of successful contact
- **SNR**: Estimated signal-to-noise ratio in dB
- **MUF**: Maximum Usable Frequency for the path
- **Distance**: Great circle distance in km
- **Bearing**: Heading from your QTH in degrees

24-Hour Timeline

Shows when each band opens to different regions throughout the day. Plan your operating around these predictions!

Propagation Model

This system uses **ITU-R based propagation calculations** including:

- Critical frequency (foF2) modeling from solar flux
- MUF calculation with path geometry
- Multi-hop F-layer and single-hop E-layer paths
- Diurnal variation (day/night effects)
- Geomagnetic absorption
- Distance-dependent path loss
- Gray-line enhancement

Note: This is a sophisticated model but simplified compared to full VOACAP. For even more accuracy, you can enhance this with:

- Full VOACAP integration using `pythonprop` library
- Local ionospheric data
- Historical validation against your actual QSOs

DXCC Integration

The system automatically loads your QSO log data to show:

- Total entities worked
- LoTW confirmed count
- Most-wanted entities

To update your DXCC data:

1. Export new ADIF from QRZ or your logging software
2. Run the original `parse_adif.py` script
3. Regenerate predictions

Customization

Adding/Removing Target Regions

Edit `TARGET_REGIONS` in `generate_propagation.py`:

python

```
TARGET_REGIONS = {
    'EU': {'name': 'Europe', 'lat': 50.0, 'lon': 10.0, 'bearing': 45},
    # Add your custom regions here
}
```

Adjusting Prediction Parameters

You can tune the propagation model in the `calculate_signal_quality()` function:

- Adjust optimal F/MUF ratio (default 0.85)
- Modify absorption factors
- Change reliability thresholds

Styling

Edit the CSS in the `<style>` section of `propagation_dashboard.html` to customize:

- Color scheme
- Font sizes
- Layout grid
- Dark/light mode

Deployment Options

Option 1: Local Use

Just open the HTML file directly in your browser after generating predictions.

Option 2: Static Hosting (Free)

1. Generate predictions locally
2. Upload `propagation_dashboard.html` and `propagation_data.json` to:
    - GitHub Pages
    - Netlify
    - Cloudflare Pages
    - AWS S3 + CloudFront

Option 3: Automated Updates

Set up a cron job or GitHub Action to:

1. Run `generate_propagation.py` hourly
2. Commit updated JSON to git
3. Auto-deploy to static hosting

Example GitHub Action:


yaml

```yaml
name: Update Propagation Data
on:
  schedule:
    - cron: '0 */2 * * *'  # Every 2 hours
jobs:
  update:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v2
      - run: python3 generate_propagation.py
      - uses: stefanzweifel/git-auto-commit-action@v4
```

Data Sources

- **Solar Data**: NOAA Space Weather Prediction Center
  - https://services.swpc.noaa.gov/json/
- **DXCC List**: ARRL DXCC program (340 current entities)
- **Propagation Models**: ITU-R Recommendations

Future Enhancements

Possible additions:

- [ ] Full VOACAP integration for higher accuracy
- [ ] Gray line visualization on map
- [ ] Auroral oval display
- [ ] Solar X-ray flux monitoring
- [ ] Historical propagation analysis
- [ ] Contest-specific band predictions
- [ ] Mobile app version (PWA)
- [ ] Email/SMS alerts for rare DX openings
- [ ] Integration with DX cluster for real-time spots

Technical Details

Languages:

- Python 3.x for prediction generation
- HTML/CSS/JavaScript for display
- Leaflet.js for mapping

Browser Compatibility:

- Chrome/Edge: ✅ Fully supported
- Firefox: ✅ Fully supported
- Safari: ✅ Fully supported
- Mobile: ✅ Responsive design

Performance:

- Prediction generation: ~2-3 seconds
- Page load time: <500ms
- JSON file size: ~100KB
- No backend server required

Support

For questions or improvements, this is a custom tool built for VE1ATM's specific needs and setup.

License

Personal use for VE1ATM. Feel free to adapt for your own station!

---

73 de VE1ATM